MIPAR User Manual

v4.1 — Last update: Sep 14, 2022

MIPAR

Table of Contents

Getting Started

Welcome to the MIPAR User Manual!

This manual is meant to complement the various tutorials and 1-on-1 training sessions available on the MIPAR website. It offers an overview of the each application, and provides a comprehensive description of the Image Processor's function library.

Topics

This Section

Other Sections

Install and Activate

Documents

- Installation + Trial Activation – WINDOWS
  Describes procedures for installing, activating, and setting up MIPAR on Windows systems
- Installation + Trial Activation – MAC
  Describes procedures for installing, activating, and setting up MIPAR on Mac systems
- Local License Activation
  Describes procedures for retrieving and activating your purchased local License Key
- Network License Activation – SERVER
  Describes procedures for retrieving and activating your purchased network License Key, and for configuring your license server
- Network License Activation – VM SERVER
  Describes procedures for retrieving and activating your purchased network License Key, and for configuring your license server
- Network License Activation – CLIENT
  Describes procedure for configuring client systems on your network to use network seats
- Network License Manager README
  Detailed documentation on installing and configuring your license server, including supported operating systems.

Troubleshooting

- Installation Troubleshooting
- Activation Troubleshooting
- License Key Transfer

Installation Troubleshooting

Error	Cause	Possible Solution
MIPAR crashes on starup or during launching of an app **OR** HASP API DLL dynamically unloaded with active logins	MIPAR crashed on startup and thus did not properly log out of the license. This is a symptom rather than the cause of the crash.	**Windows**: Try updating your graphics card driver (see below links). If the problem persists, try the solution below. NVIDIA: https://www.nvidia.com/download/index.aspx?lang=en-us AMD: https://www.amd.com/en/support Intel: https://downloadcenter.intel.com/product/80939/Graphics
Message on Mac stating that “MIPAR Installer cannot be checked for malicious software”.	macOS is not recognizing MIPAR Software as an identified developer	- Right-click on MIPAR Installer - Click “Open” - Click “Open” in the message that appears
Permissions setting error during installation stating that “‘Icacls’ is not recognized as an internal command, operable program or batch file.”	Incorrect PATH variable	- Open System Properties > Advanced tab > Environment Variables > System variables - Make sure the Path variable contains the entries “C:\Windows\system32” and %SystemRoot%\system32 - Retry running MIPAR Installer
On startup error: Failed to set up java output streams	Runtime environment was not installed properly	- Uninstall MRE (MATLAB Runtime 9.11) - Download and run installer from Link - Restart the system and MIPAR
On startup error: Could not access the MATLAB runtime component cache.	Permission issue accessing runtime environment	- Launch MIPAR as administrator by right-clicking > Run as administrator If problem persists: - Uninstall MRE (MATLAB Runtime 9.11)

		- Download and run installer from Link - Restart the system and MIPAR
Interface buttons and text display incoherent text	Fonts were not installed properly	- Close MIPAR - Navigate to C:\ProgramFiles\MIPAR\fonts - Select all .ttf and .otf files - Right click > Install (or “Install for all users” if available) - Launch MIPAR
MIPAR takes an extremely long time to launch on a system with an NVIDIA Ampere architecture video card installed	MIPAR must recompile CUDA drivers on launch	Add Environmental variable: CUDA_CACHE_MAXSIZE with a variable value: 536870912

Solution for Crash on Startup or “HASP API DLL dynamically unloaded” Message (Windows)

This error occurs when the MIPAR Runtime crashes on boot. Please try in order:

Delete the MCR Cache

- Navigate to the MIPAR install folder, default: C:\Program Files\MIPAR\
- Delete the folder: mcrCache9.11
- Restart Computer
- If this does not work, try:

Delete Preferences

- Navigate to C:\Program Files\MIPAR (installation path)
- Delete miparPreferences.pref
- Retry MIPAR launch
- If this does not work, try:

Reinstall the Runtime

- Uninstall the runtime from the system: MATLAB Runtime 9.11
- Download and install the runtime: LINK
- Restart the computer
- If this does not work, try:

Adjust Permissions

- Try launching MIPAR as an admin
- Are you running an antivirus program? Does whitelisting MIPAR resolve the problem?

Activation Troubleshooting

Error	During	Cause	Possible Solution
MIPAR crashes on starup or during launching of an app **OR** HASP API DLL dynamically unloaded with active logins	Startup	MIPAR crashed on startup and thus did not properly log out of the license. This is a symptom rather than the cause of the crash.	Try updating your graphics card driver (see below links). If the problem persists, try launching MIPAR as Administrator. If the problem still remains, try uninstalling and reinstalling MIPAR NVIDIA: https://www.nvidia.com/download/index.aspx?lang=en-us AMD: https://www.amd.com/en/support Intel: https://downloadcenter.intel.com/product/80939/Graphics
License Manager not reachable **OR** Unknown Error	Activation	License Manager may not have been installed properly **OR** License Manager may be out-of-date	Try accessing http://localhost:1947/_int_/ACC_help_index.html. **If this page cannot be reached:** - See Solution for License Manager Not Reachable below. **If this page can be reached:** **WINDOWS** - Update to the latest License Manager version for Windows - If this still does not resolve the issue, try activating the license manually: Quit MIPAR > Go to http://localhost:1947/_int_/checkin.html > Click “Choose File” > choose .v2c license key file > Click “Apply File” > Launch MIPAR **MACOS** - Update to the latest License Manager version for Mac -If this still does not resolve the issue, try activating the license manually: Quit MIPAR > Go to http://localhost:1947/_int_/checkin.html > Click “Choose File” > choose .v2c license key file > Click “Apply File” > Launch MIPAR

Message on Mac stating that “MIPAR cannot be checked for malicious software”.	Activation	macOS is not recognizing MIPAR Software as an identified developer	– Right-click on MIPAR - Click “Open” - Click “Open” in the message that appears
Cannot select .v2c file on Mac	Activation	.v2c file was saved with a .xml extension and cannot be selected after clicking “Activate”	First remove the .xml extension and retry activation. If this does not work, open the file in Text Edit and resave it with only the .v2c extension. When saving, make sure to uncheck the box which says “Add .txt if no extension is present”
HASP_UPDATE_ALREADY_ADDED	Activation	This MIPAR License has already been activated previously.	If you have not requested a trial license, you may do so here. Visit our Purchase Page to obtain a quote for local or network licenses.
HASP_DEVICE_ERR	Activation	An issue which affects some Mac systems running macOS 10.13 High Sierra, or on Windows systems could be caused by corrupted secure storage.	See Solution for HASP_DEVICE_ERR below
HASP_CLONE_DETECTED **OR** “Cloned” / “Secure Storage ID Mismatch” shown next to some keys	Activation	A change in your system’s hardware has caused the license to be	See Solution for HASP_CLONE_DETECTED below

		temporarily deactivated for clone protection reasons.	
HASP_NO_VLIB	Activation	The vendor library required to activate MIPAR licenses was not properly copied over during the installation.	See Solution for HASP_NO_VLIB below
HASP_HASP_NOT_FOUND	Activation	A MIPAR License has not been activated, or cannot be found on your network (for network licenses).	Please be sure to follow the instructions in the Activation Window (which appears after MIPAR is launched) to activate your license.
HASP_FEATURE_NOT_FOUND	Activation	A MIPAR License has not been activated, or cannot be found on your network (for network licenses).	Please be sure to follow the instructions in the Activation Window (which appears after MIPAR is launched) to activate your license.
HASP_FEATURE_EXPIRED	Activation	Your MIPAR License has expired.	If you have not requested a trial license, you may do so here. Visit our Purchase Page to obtain a quote for local or network licenses.
HASP_INV_UPDATE_DATA	Activation	Your .v2c file may have been corrupted during	Please contact support@mipar.us for assistance.

		download.	
HASP_TOO_MANY_USERS	Activation	The concurrent user limit on your network license has been reached.	You'll need to wait for another user to close MIPAR. Contact support@mipar.us to inquire about obtaining additional seats for your network license.
Java Exception Occurred: Sentinel EMS is not available for the specified URL	Activation	Sentinel license manager is not able to communicate through the firewall.	Add a firewall rule to allow connection for the Sentinel license managed (this can be disabled after activation) Instructions: LINK HASP executable: C:\Program Files (x86)\Common Files\Aladdin Shared\HASP\hasplms.exe

Solution for License Manager Not Reachable

Windows

Check License Manager Service

1. In the Windows search bar type **Services**
2. Open the **Services** app
3. Scroll down to **Sentinel LDK License Manager**
4. If it exists and is stopped, select it and click **Start** in the left panel to start the service
5. If it does not exist, proceed with steps 6-15

Set Environment Variables

6. Open Command Prompt as Administrator and execute the following commands:
7. MKDIR c:\ldktemp
8. SET TEMP=C:\ldktemp
9. SET TMP=C:\ldktemp

Repair License Manager

10. Download the License Manager package for Windows
11. Extract the downloaded **Install_License_Manager_Win.zip** (Right-click > Extract All)
12. Inside the extracted folder, double-click **Install License Manager.bat**
13. Retry activation through MIPAR's Activation Window

Mac

Repair License Manager

1. Download the License Manager package for Mac
2. Double-click the downloaded **Install_License_Manager_Mac.pkg**
3. Follow the prompts to complete the License Manager update

Add Vendor Library

4. Download the MIPAR Vendor Library for Mac
5. From menu bar in Finder, select Go > Go to folder > type **/var/hasplm** and press **Enter**
6. Place the downloaded **haspvlib_111047. dylib** file here

Allow Vendor Library

7. Right-click on the **haspvlib_111047. dylib** file > **Open**
8. If you see a message which states that “haspvlib_111047 is from an unidentified developer…”, choose **Open**
9. Quit Terminal if it opens
10. If there was no “Open” button in the previous message, go to System Preferences > Security & Privacy > General > click Allow next to the message toward the bottom

Restart License Manager

Restart License Manager

11. In Spotlight search bar type **Terminal**
12. Open **Terminal** app
13. Type **sudo launchctl unload /Library/LaunchDaemons/com.aladdin.hasplmd.plist** and press **Enter**
14. Type **sudo launchctl load /Library/LaunchDaemons/com.aladdin.hasplmd.plist** and press **Enter**

Solution for HASP_DEVICE_ERR

Windows

Repair License Manager

1. Download the License Manager package for Windows
2. Extract the downloaded **Install_License_Manager_Win.zip** (Right-click > Extract All)
3. Inside the extracted folder, double-click **Install License Manager.bat**

Mac

Repair License Manager

1. Download the License Manager package for Mac
2. Double-click the downloaded **Install_License_Manager_Mac.pkg**
3. Follow the prompts to complete the License Manager update
4. After the installer finishes, and the green check mark is shown, leave the installer window open and proceed with steps 5-6

Allow in System Preferences

5. Go to System Preferences > Security & Privacy
6. Click the “Allow” button next to “SFNT Germany GmbH” (close the installer window and try clicking “Allow” again if it does not respond at first)

Solution for HASP_CLONE_DETECTED

Check for Cloned Key

1. On your MIPAR system, go to http://localhost:1947/_int_/devices.html
2. If there is a key marked with a red “Cloned” or “Secure Storage ID Mismatch”, click the “C2V” button in its “Actions” column, and send the C2V file to support@mipar.us. If there are no visible keys, proceed with steps 3-6.

Repair License Manager

For Windows

3. Download the License Manager package for Windows
4. Extract the downloaded **update_license_manager_win.zip** (Right-click > Extract All)
5. Inside the extracted folder, double-click **Update License Manager.bat**

For Mac

3. Download the License Manager package for Mac
4. Double-click the downloaded **Install_License_Manager_Mac.pkg** > Follow the prompts to complete the License Manager update
5. Go to http://localhost:1947/_int_/devices.html and see if any keys show up

6. Locate the key marked with red “Cloned”, click the “C2V” button under its “Actions” column, and send the C2V file to support@mipar.us. If there are still no visible keys marked “Cloned”, please contact support@mipar.us for further assistance.

Solution for HASP_NO_VLIB

Windows

Add Vendor Library

1. Download the MIPAR Vendor Library for Windows
2. In Windows Explorer, navigate to **C:\Program Files (x86)\Common Files\Aladdin Shared\HASP**
3. Place the downloaded **haspvlib_111047.dll** file here

Restart License Manager

4. In the Windows search bar type **Services**
5. Open the **Services** app
6. Scroll down to **Sentinel LDK License Manager**
7. Select it and click **Restart** in the left panel

Mac

Add Vendor Library

1. Download the MIPAR Vendor Library for Mac
2. From the menu bar in Finder, select Go > Go to folder > type **/var/hasplm** and press **Enter**
3. Place the downloaded **haspvlib_111047. dylib** file here

Allow Vendor Library

4. Right-click on the **haspvlib_111047. dylib** file > **Open**
5. If you see a message which states that “haspvlib_111047 is from an unidentified developer…”, choose **Open**
6. Quit Terminal if it opens
7. If there was no “Open” button in the previous message, go to System Preferences > Security & Privacy > General > click Allow next to the message toward the bottom

Restart License Manager

Restart License Manager

8. In Spotlight search bar type **Terminal**
9. Open **Terminal** app
10. Type **sudo launchctl unload /Library/LaunchDaemons/com.aladdin.hasplmd.plist** and press **Enter**
11. Type **sudo launchctl load /Library/LaunchDaemons/com.aladdin.hasplmd.plist** and press **Enter**

License Key Transfer

Please contact support@mipar.us with your request to transfer a license. A representative will be in contact shortly after to assist you.

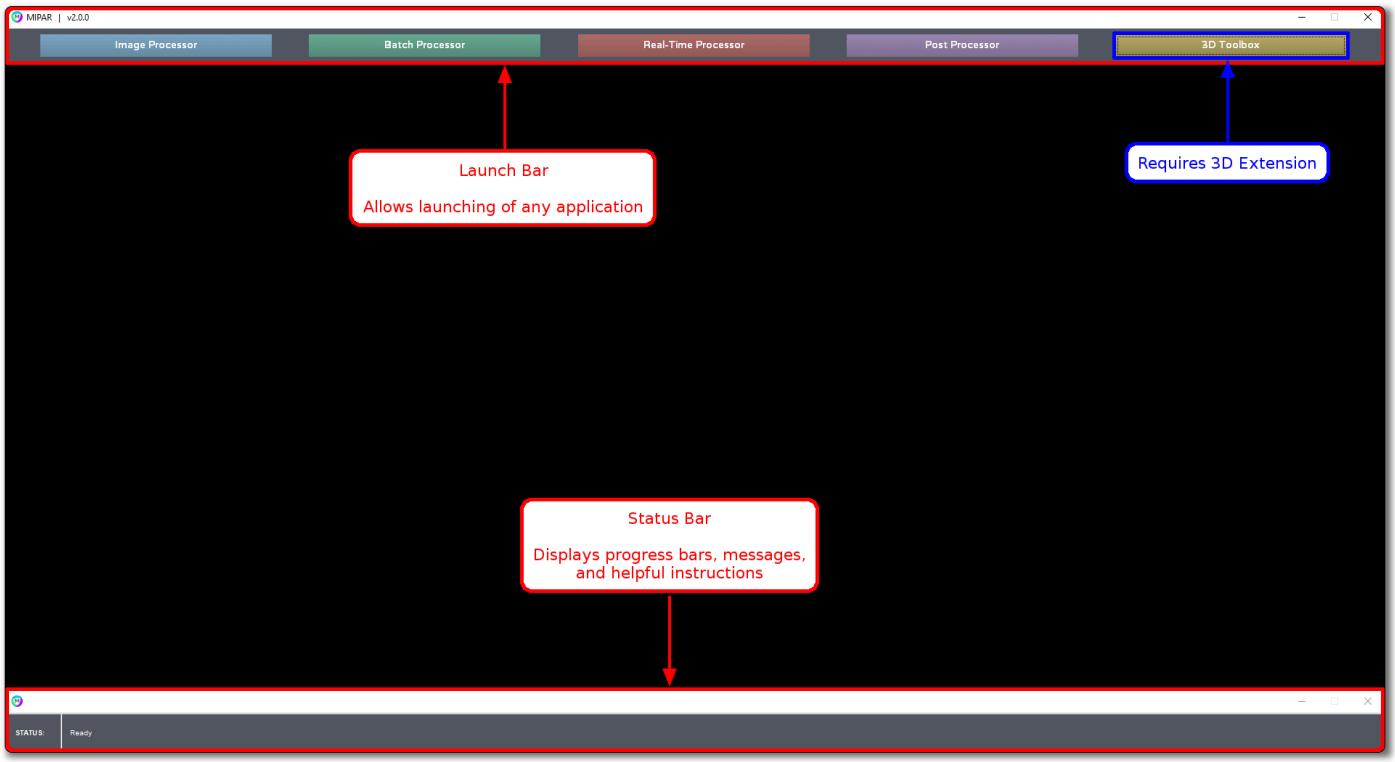
License transfers are only available to users with an active Maintenance Plan, and are limited to 3 per year.

Introduction

MIPAR is a revolutionary image analysis software, capable of identifying and measuring features from nearly any image one can capture. Our users have shown it to be perhaps the most the efficient and flexible image processing software on the planet.

Through five integrated applications, MIPAR offers powerful and efficient environments for the different tasks performed during 2D and 3D image analysis. Developed by materials scientists, it is uniquely designed to offer workflows that are well-suited to solve a variety of scientific image analysis problems.

Launching MIPAR



The Apps

- Image Processor
  The most commonly used app. Used for building and editing Recipes, as well as applying Recipes to single images. Recipes are sequences of image processing steps which work to identify features of interest from your image.

- Batch Processor
  Used for batch applying a Recipe to multiple images. A batch process will save the last Recipe step (or each Layer) for each image, along with a spreadsheet of global measurements, if there were measurements in the Recipe.
  With 3D Extension: The Batch Processor can output Image Stacksand 3D Reconstructions

- Real-Time Processor
  Similar to the Batch Processor, but used for applying a Recipe to multiple images, every time a new image is added to a folder being “watched”. This is typically a folder on a network storage drive, such that images may be processed in real-time as they acquired and saved from an imaging instrument. *With 3D Extension:* The Real-Time Processor can output Image Stacks and 3D Reconstructions

- Post Processor
  Used for reviewing the results of a batch or real-time process. All saved processed images are loaded and overlaid/outlined, etc. on their respective references. The user can flip through each processed result to assess accuracy. Manual edits can be made to correct errors. Global and feature measurements can be made from all processed images at once.

- 3D Toolbox *(requires 3D Extension)*
  Used for visualizing and editing Image Stacks, as well as visualizing, editing, and quantifying 3D Reconstructions. Reconstructions can be exported to other visualization applications such as commercial Avizo open source ParaView and for high-end rendering and animation.

Supported Formats

2D Formats

The following 2D image formats are supported for use in the Image Processor, Batch Processor, Real-Time Processor, and Processed Image Editor:

- TIF
- JPG
- BMP
- PNG
- GIF
- Bio-Formats (over 150 formats)
  Requires downloading the Bio-Formats package and placing inside the "plugins" folder in your installation directory

> As of v4.0.0.0, MIPAR now prompts users to download a newer version of the Bio-Formats library which patches the log4j vulnerability, when opening speciality formats for the first time. Users currently running the older version will be prompted to update on launch. We strongly recommend updating. To update manually, download the new library version, place the .jar file in [Install Directory]/plugins and restart MIPAR.

> MIPAR scales images to 8-bit on load. MIPAR by default will auto contrast higher bit images, to turn off auto contrast on load (File>Preferences>Normalize Image Before Scaling)

3D Formats

MIPAR creates 3D Reconstruction files from a series of 2D images through use of the Batch Processor or Real-Time Processor. 3D Reconstruction files are saved from the Batch Processor or Real-Time Processor in MIPAR's REC format and can be opened in the 3D Toolbox.

System Requirements

Supported Operating Systems

Windows (64-bit only)

- Windows 11
- Windows 10
- Windows 7 SP 1
- Windows Server 2019
- Windows Server 2016

> **Note:** There is a known problem with Windows 7 SP1 build 7601 ran on Intel® Core™ i9-7900X CPU in MIPAR version 3.x.x that might cause the software to crash.

Mac

- macOS Monterey (12.0)
- macOS Big Sur (11.0)
- macOS Catalina (10.15.6 or later)

> **Note:** Apps such as BetterTouchTool, or Magnet can affect MIPAR performance on Mac. If you use these apps, we recommend disabling them while using MIPAR.

> **Note:** MIPAR is not yet optimized for Apple Silicon Macs, but will run under Rosetta 2. Full Apple Silicon support is planned for a future release.

Supported Languages

MIPAR's user-interface is only available in English, but it can be installed on all languages of Windows and macOS.

Minimum System Requirements

- **CPU:** Dual-core modern Intel / AMD
- **GPU:** Intel HD 4000 level graphics
- **Storage:** HDD 5400 RPM
- **RAM:** 4 GB

Recommended System Specs

- **CPU:** 8+ core modern Intel / AMD
- **GPU:** NVIDIA GeForce GTX 1060 or better
- **Storage:** SSD (Solid State Drive)
- **RAM:** 16+ GB

Deep Learning Requirements

GPU Computation

Enable

From any app, launch *File* > *Preferences*. In the GPU panel, set to “GPU Computation” to “On” if you have a supported GPU. Click here to learn more about the technical aspects and benefits of GPU computation.

GPU computation requires an NVIDIA GPU with **compute capability 3.5 or higher** (Kepler, Maxwell, Pascal, Volta, Turing, or Ampere architecture), and your driver must support **CUDA Toolkit 11 or higher**. We recommend you update your NVIDIA driver to the latest version here. Learn more about CUDA here.

For GPU requirements for deep learning training, see Deep Learning System Requirements.

Some functions may produce slightly different results when run on the GPU vs. the CPU.

Supported Functions

The following functions support GPU Computation:

Edit

- Crop Image
- Resize Image
- Rotate Image
- Flip Image
- Translate Image
- *Register Image: Free-Form, Translation, Rigid, Affine
- Sparsely Sample Image
- Draw Random Lines

Color

- Color Cluster
- Color Deconvolution
- Channel Operation

Pre-Processing

- Adjust Contrast
- Flatten Background
- Smart Cluster

- Wiener Filter
- Gaussian Blur
- Average Blur
- Sum Filter
- Grayscale Dilate
- Grayscale Erode
- StdDev Filter
- Gradient Filter
- Highlight Lines
- Bright Texture
- Dark Texture
- Advanced Texture
- Pattern Mapping
- Orthogonal Correlate
- *Cross-Correlate
- Sharpen
- FFT Filter
- Make FFT
- Apply Stored FFT Filter
- *Grayscale Interpolation

Segmentation

- Invert
- Blank
- Basic Threshold
- Range Threshold
- Adaptive Threshold
- E-M Threshold
- *Local Threshold
- Find Edges
- Find Circles
- Find Lines
- Find Text
- Find Facial Features
- *Region Grow
- Find Global Maximum
- Find Global Minimum
- Find Local Maxima
- Find Local Minima

Deep Learning

- Apply Model
- Call Output

Morphology

- Uniform Dilation
- Smart Dilation
- Retain Dilation
- Uniform Erosion
- Smart Erosion
- Retain Erosion
- Smooth Features
- Extend Features
- Perimeter Pixels
- Skeletonization
- Thin
- Shrink
- Distance Map

Clean-Up

- Relative Size Filter
- Replace With: Centroid, Ellipse, Equivalent Circle, Fit Circle, Largest Circle, Local Normal, Major Axis, Minor Axis, Nearest Distance, Triangulation
- Mark Center

Memory

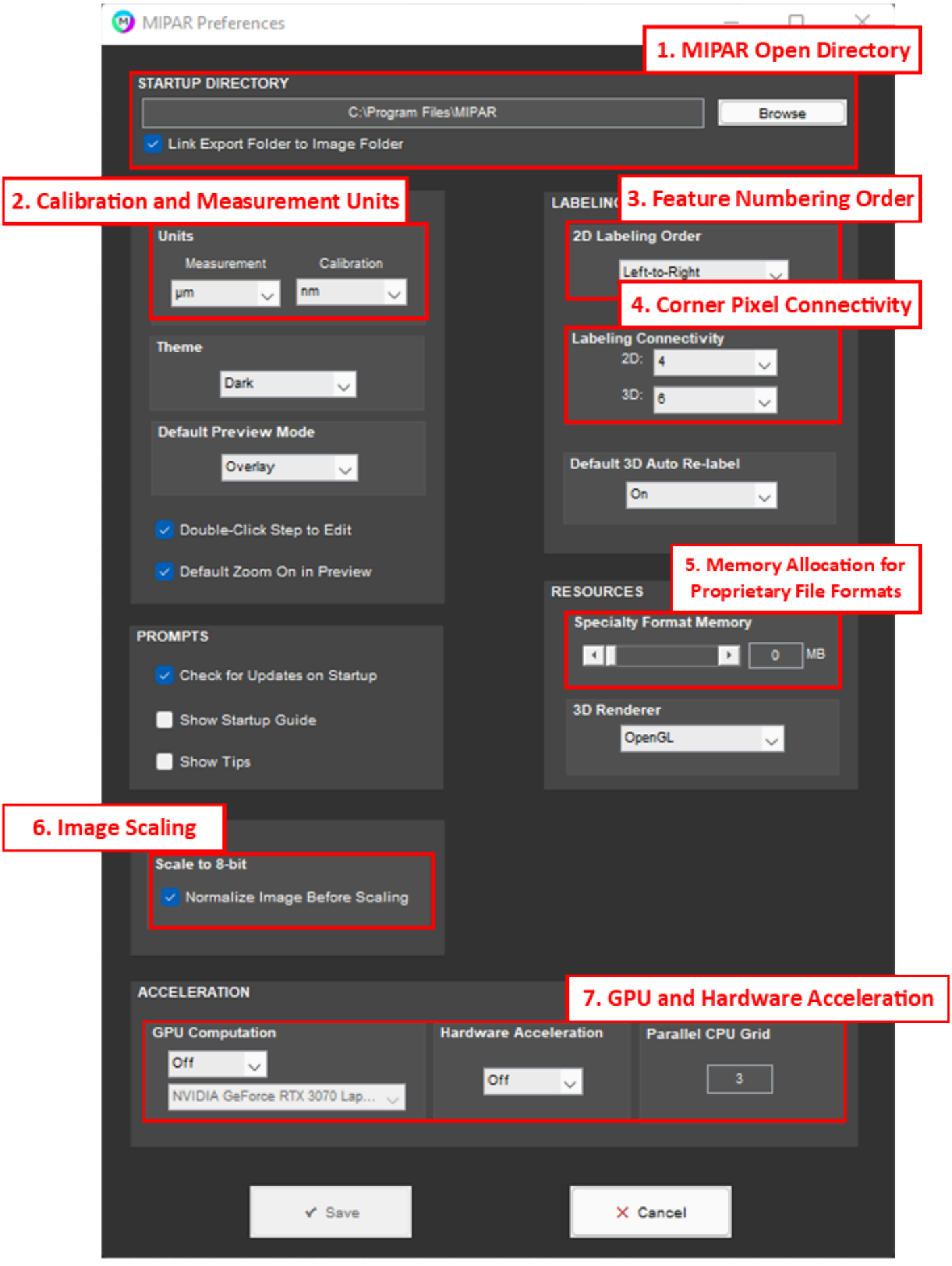
- Set Companion Image
- Load Companion Image
- Call Companion Image
- Set Memory Image #1-6
- Call Memory Image #1-6
- Call Original Image

Math

- *Feature Union
- *Feature Minus
- *Feature Intersection
- *Keep Exclusive Features
- Make Grayscale
- *Add
- *Average
- *Divide
- *Multiply
- *Subtract
- Add Value
- *Merge Darker Pixels
- *Merge Lighter Pixels

Preferences

Layout


MIPAR Preferences
1. MIPAR Open Directory
STARTUP DIRECTORY
C:\Program Files\MIPAR
Browse
Link Export Folder to Image Folder
2. Calibration and Measurement Units
Units
Measurement
Calibration
µm
nm
Theme
Dark
Default Preview Mode
Overlay
Double-Click Step to Edit
Default Zoom On in Preview
3. Feature Numbering Order
2D Labeling Order
Left-to-Right
4. Corner Pixel Connectivity
Labeling Connectivity
2D: 4
3D: 6
Default 3D Auto Re-label
On
5. Memory Allocation for Proprietary File Formats
RESOURCES
Specialty Format Memory
0
MB
3D Renderer
OpenGL
PROMPTS
Check for Updates on Startup
Show Startup Guide
Show Tips
6. Image Scaling
Scale to 8-bit
Normalize Image Before Scaling
7. GPU and Hardware Acceleration
ACCELERATION
GPU Computation
Off
NVIDIA GeForce RTX 3070 Lap...
Hardware Acceleration
Off
Parallel CPU Grid
3
Save
Cancel

1. MIPAR Open Directory

Default folder for opening and image or recipe on MIPAR launch.

2. Calibration and Measurements Units

Calibration units are used to display the scale bar and are used as default during recipe calibration. Measurement units are used for measurement unit generation.

3. Feature Numbering Order

MIPAR numbers features during Reject Features function and during feature measurement generation. Left to right numbering will number features from the left edge of the image to the right. Top to Bottom numbering will number features from the top edge of the image to the bottom.

4. Corner Pixel Connectivity

Distinct features can be identified using two different methods. 4-pixel connectivity and 8-pixel connectivity.

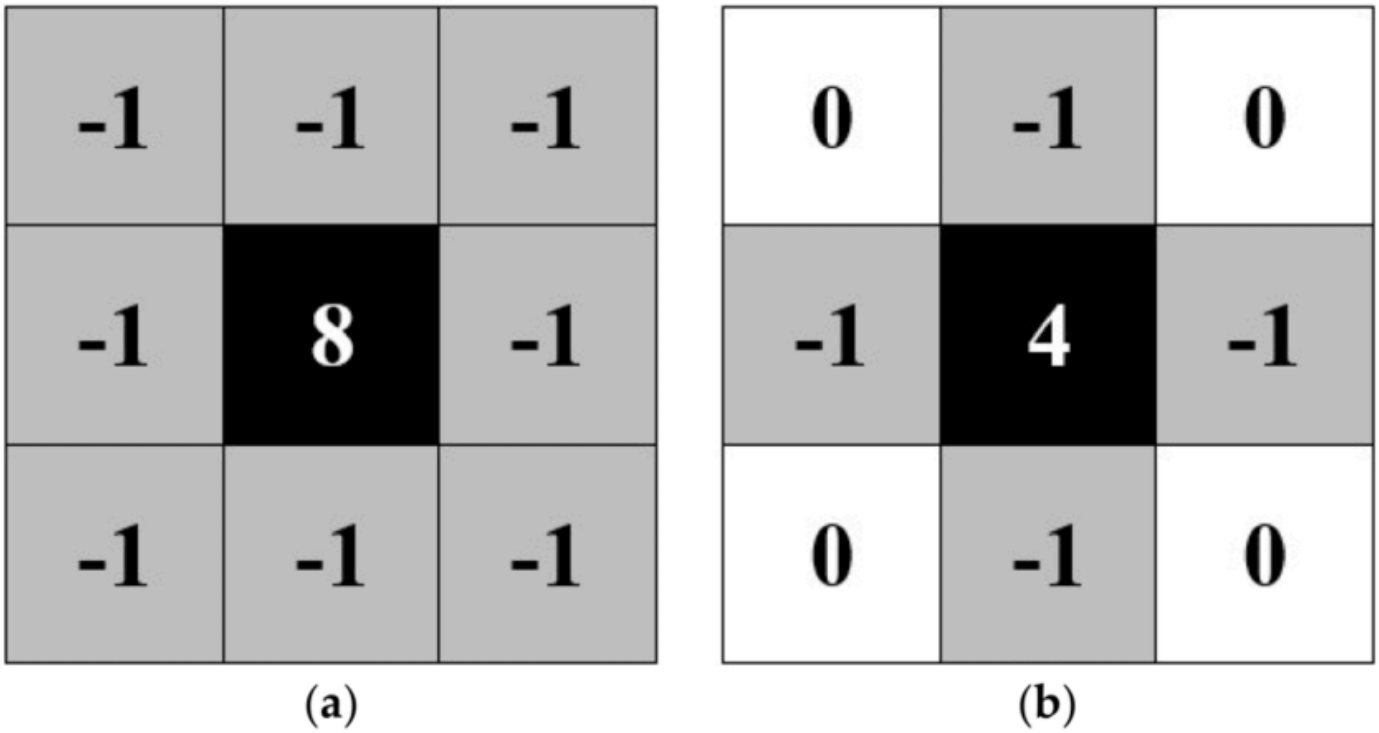


Image Source

5. Memory Allocation for Proprietary File Formats

MIPAR is able to open proprietary file types which are handled using a dedicated library. If MIPAR throws out of memory errors when the workstation is not out of memory, adjust this setting to allocate more memory

to MIPAR.

6. Image Scaling

MIPAR scales all opened images to 8-bit, scaling can be done by normalizing the images to the min and max values. We recommend disabling this setting for projects that requires intensity measurements.

7. GPU and Hardware Acceleration

GPU Computation is recommended for large images to accelerate image processing. Note that GPU selection for applying Deep Learning models is a separate setting that overrides this preference in the Apply DL Model function.

Hardware Acceleration is supported on some CPUs to improve MIPAR interface performance.

Parallel CPU Grid is used for some functions to accelerate processing. This value defaults to half the number of physical cores on the system.

Keyboard Shortcuts

On Mac systems, use CMD instead of CTRL.

All Windows

Zoom	CTRL+D
Pan	CTRL+F
Toggle Zoom/Pan	CTRL
Measure (if appl.)	CTRL+M

Recipe Step Preview Windows

Toggle Preview	SHIFT
Cycle UI Elements	TAB / SHIFT+TAB
Navigate Radio Buttons	UP / DOWN ARROWS
Accept Text Entry	ENTER
Accept Window	SHIFT+ENTER
Cancel Window	ESC

Windows with Manual Editing Tools

Switch Editing Tools	NUMBER KEYS (1, 2, 3…)
Fill Mode	CTRL+A
Erase Mode	CTRL+S
Undo	CTRL+Z

Image Processor

Open Image	CTRL+O
Load Recipe	CTRL+L
Save Current Image	CTRL+S
Save Reference Image	CTRL+A

Save Recipe	CTRL+R
Undo	CTRL+Z

Session Processor

Load Session	CTRL+L
Save Session	CTRL+E
Save Current Image	CTRL+S
Save All Images	CTRL+A
Manual Editing	See above table
Change Current Image	LEFT/RIGHT
Change Active Layer	UP/DOWN

3D Toolbox

Open Image Stack	CTRL+I
Open Reconstruction	CTRL+O
Save Stack	CTRL+T
Save Recon	CTRL+S
Undo Last Stack Change	CTRL+X
Undo Last Recon Change	CTRL+Z
Rotate 3D	CTRL+G

Tutorials

Watch video tutorials on how to use many of MIPAR's features:

- Overview
- Image Processor
- Batch Processor
- Session Processor
- AI Session Processor
- 3D Toolbox

Overview

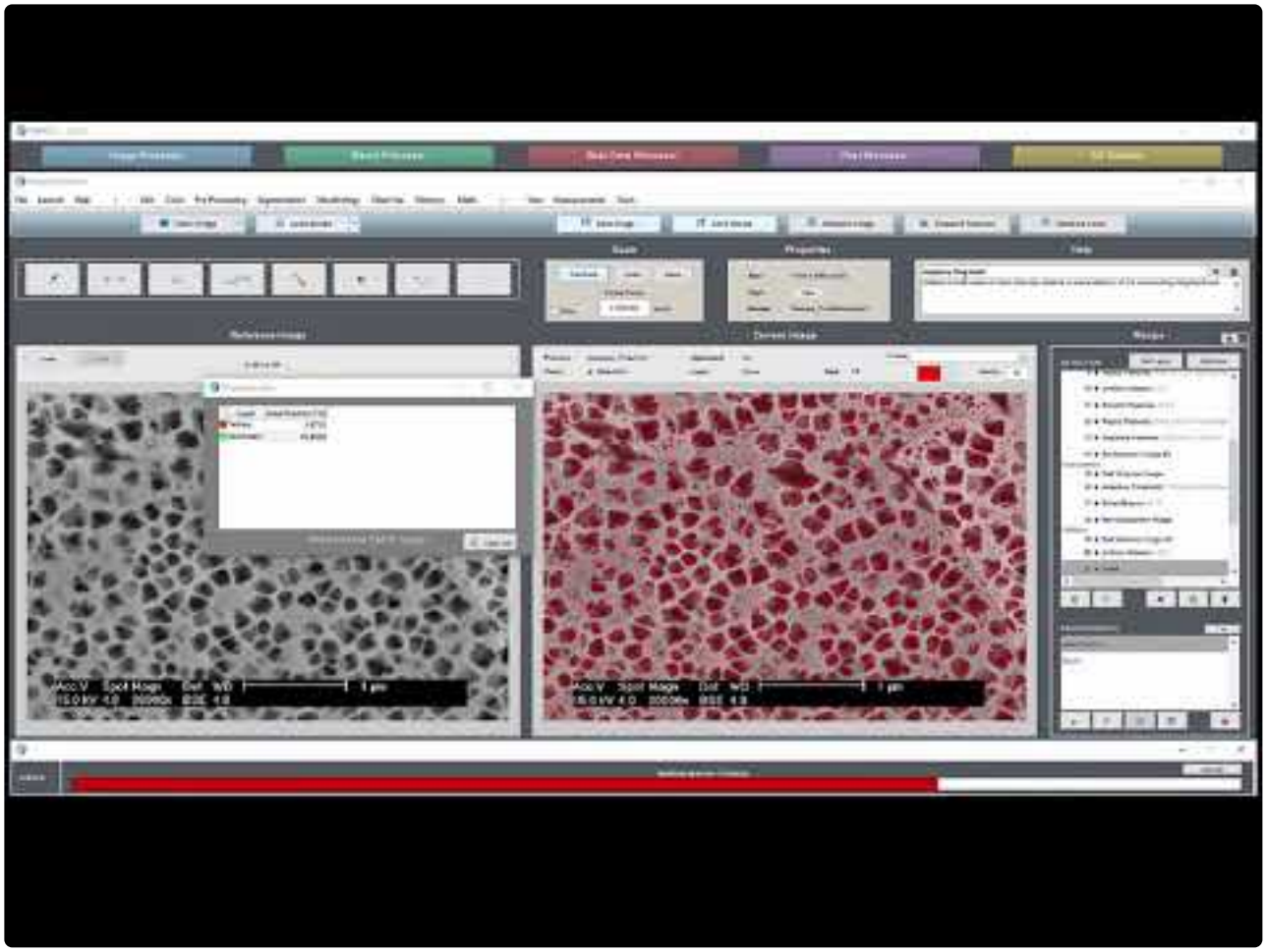
Tour

Take a tour of MIPAR's powerful features and intuitive workflow. You'll see examples of feature detection, batch processing, and measurements.

https://www.youtube.com/embed/ws-AQkPSgOw?rel=0

Demo

Watch a full demo of MIPAR's operation including the Image Processor, Batch Processor, and Post Processor apps.



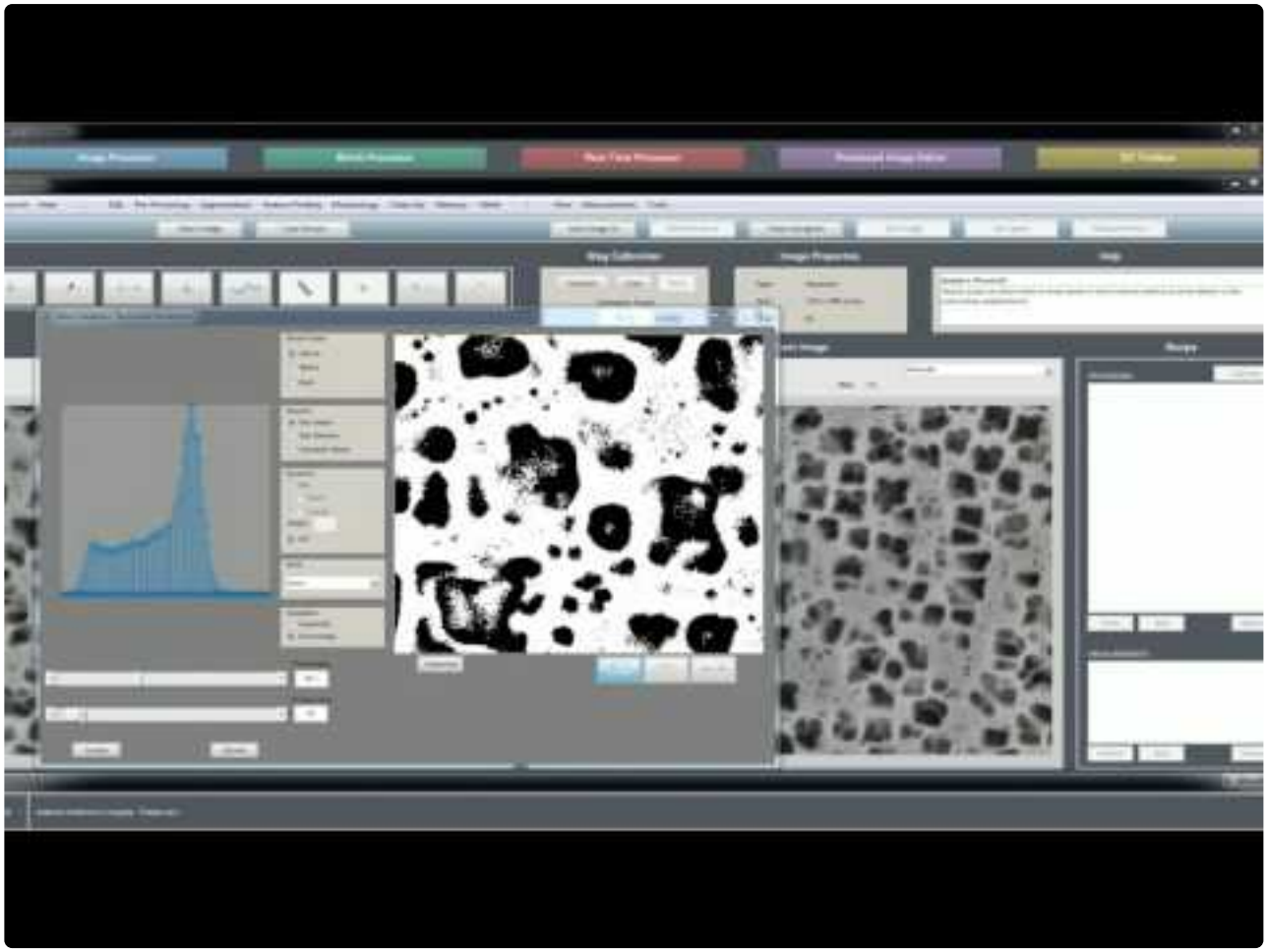
https://www.youtube.com/embed/D9vF77EB4hY?rel=0

Image Processor

Detecting

Creating a Recipe

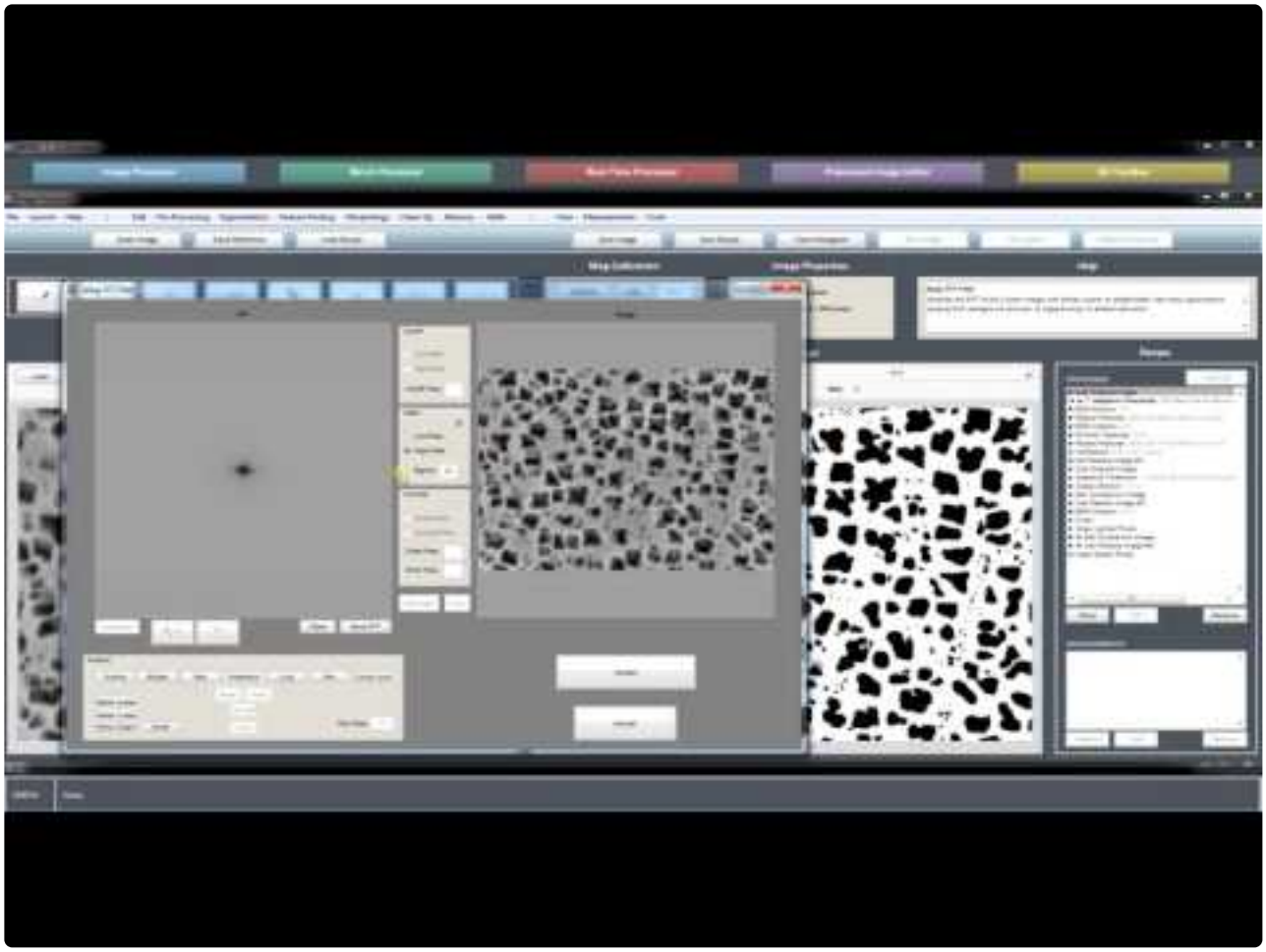
Describes the process of creating a segmentation recipe to identify features of interest in an image.



https://www.youtube.com/embed/D01HYV8yAiw?rel=0

Optimizing Steps

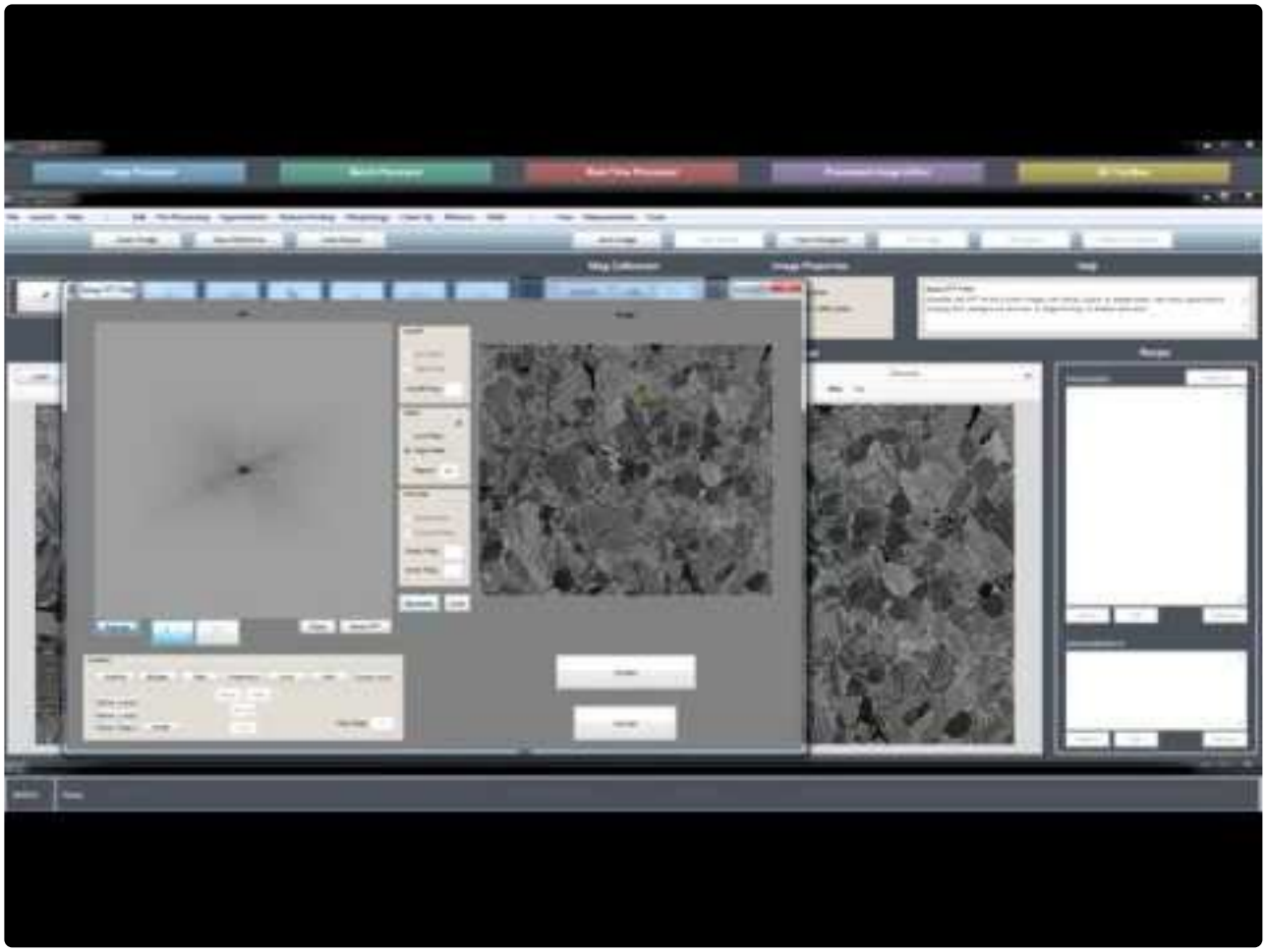
Shows the procedures and applications for optimizing, or objectively determining, image processing parameters.



https://www.youtube.com/embed/HClFGOyt6jI?rel=0

Frequency Filtering

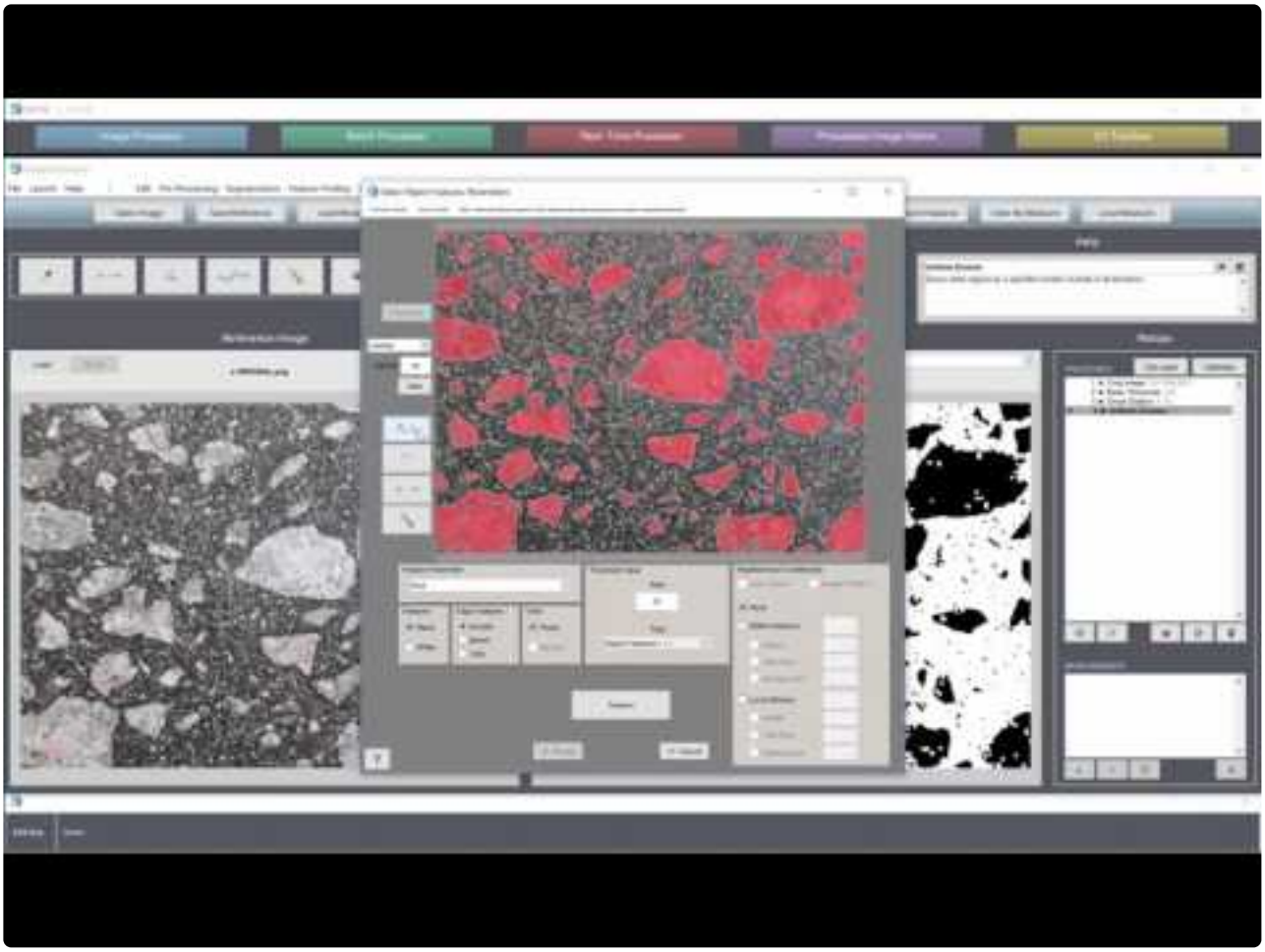
Demonstrates how to use the *Frequency Filter* in three different applications.



https://www.youtube.com/embed/X2GzqYL_i88?rel=0

Memory Steps

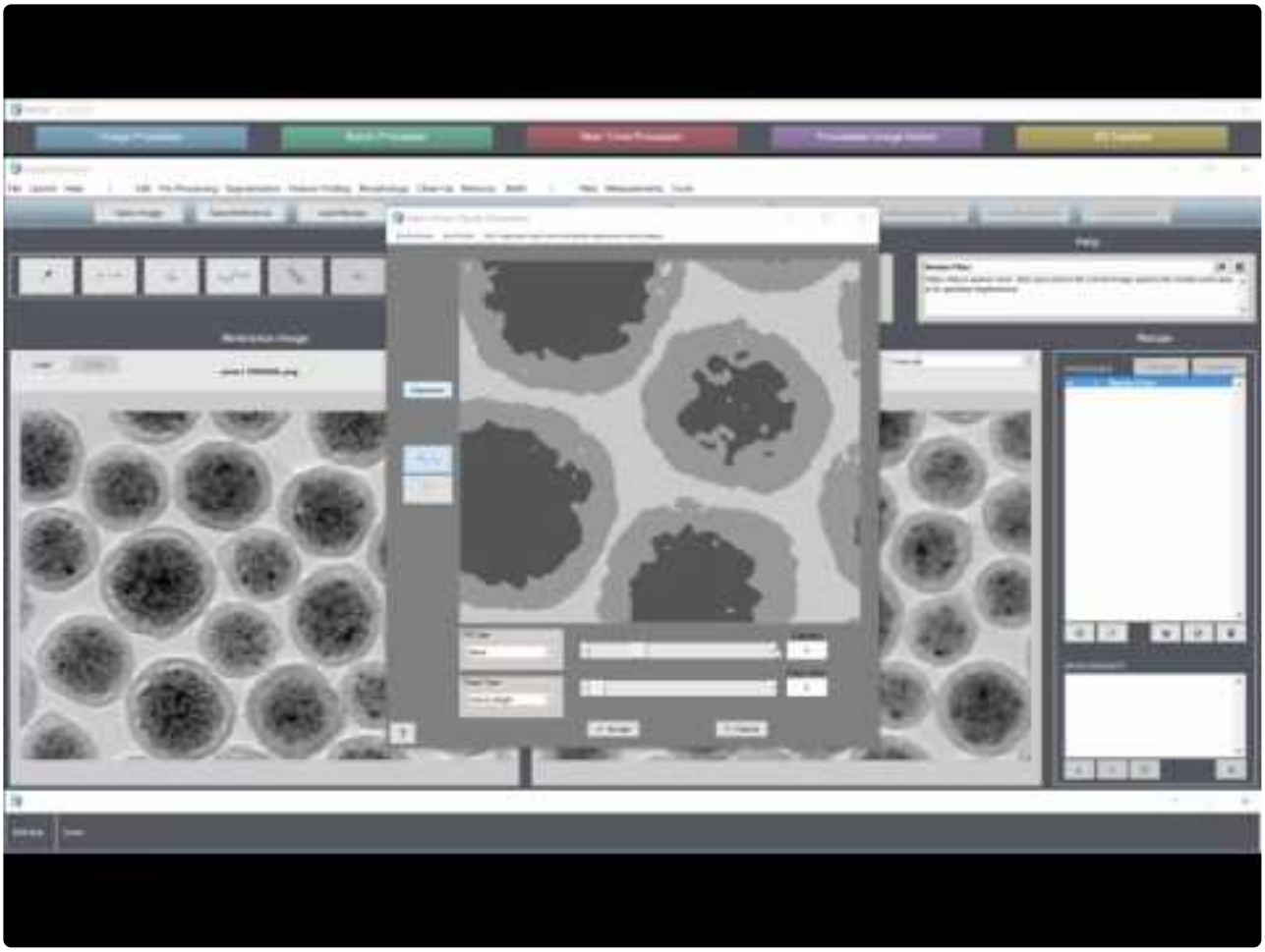
Shows examples of setting, calling, and using memory steps in a Recipe.



https://www.youtube.com/embed/ex7UHo7tvro?rel=0

Smart Cluster™

Demonstrates use of the *Smart Cluster* function for simple and objective multi-class segmentation.



https://www.youtube.com/embed/FxebQbpdRuE?rel=0

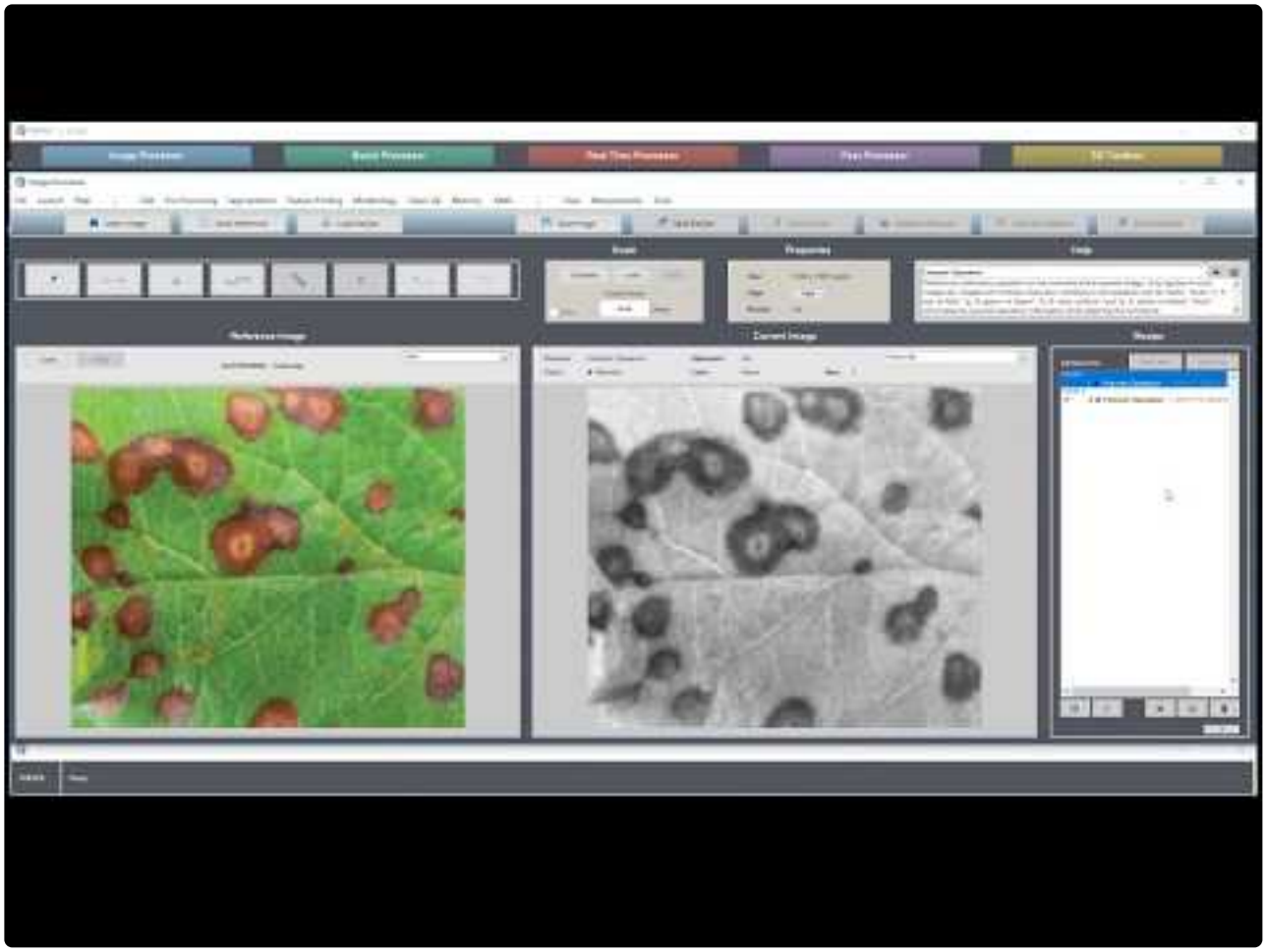
Color Cluster™

Demonstrates use of the *Color Cluster* function for simple and objective color segmentation.

https://www.youtube.com/embed/xdrw2Utgb2c?rel=0

Color Deconvolution

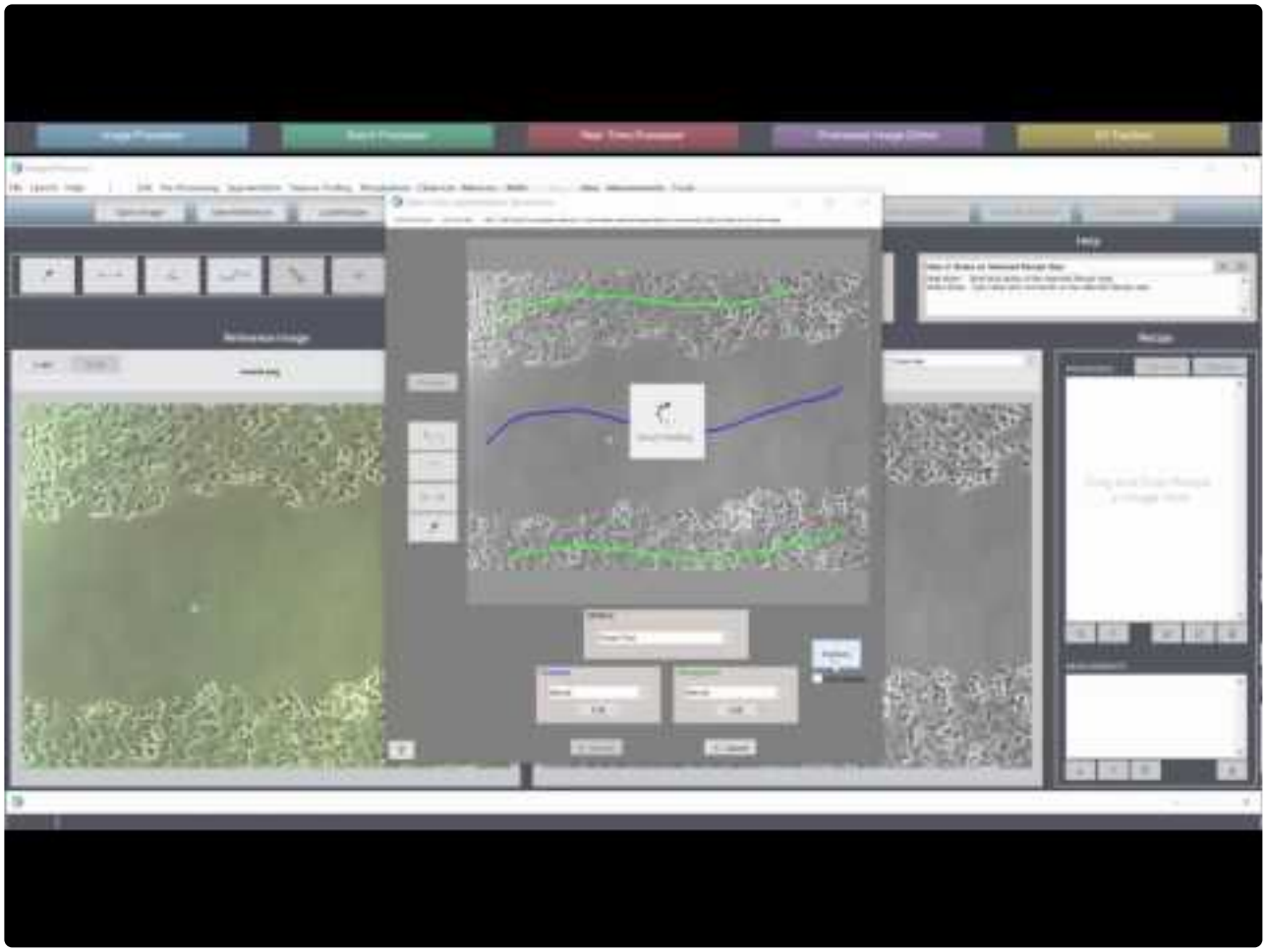
Demonstrates use of the *Color Deconvolution* function to highlight features of certain colors.



https://www.youtube.com/embed/KlkunloiWkE?rel=0

Smart Find™

Demonstrates two example uses of Smart Find in *Auto Segmentation* for powerful and objective feature detection.

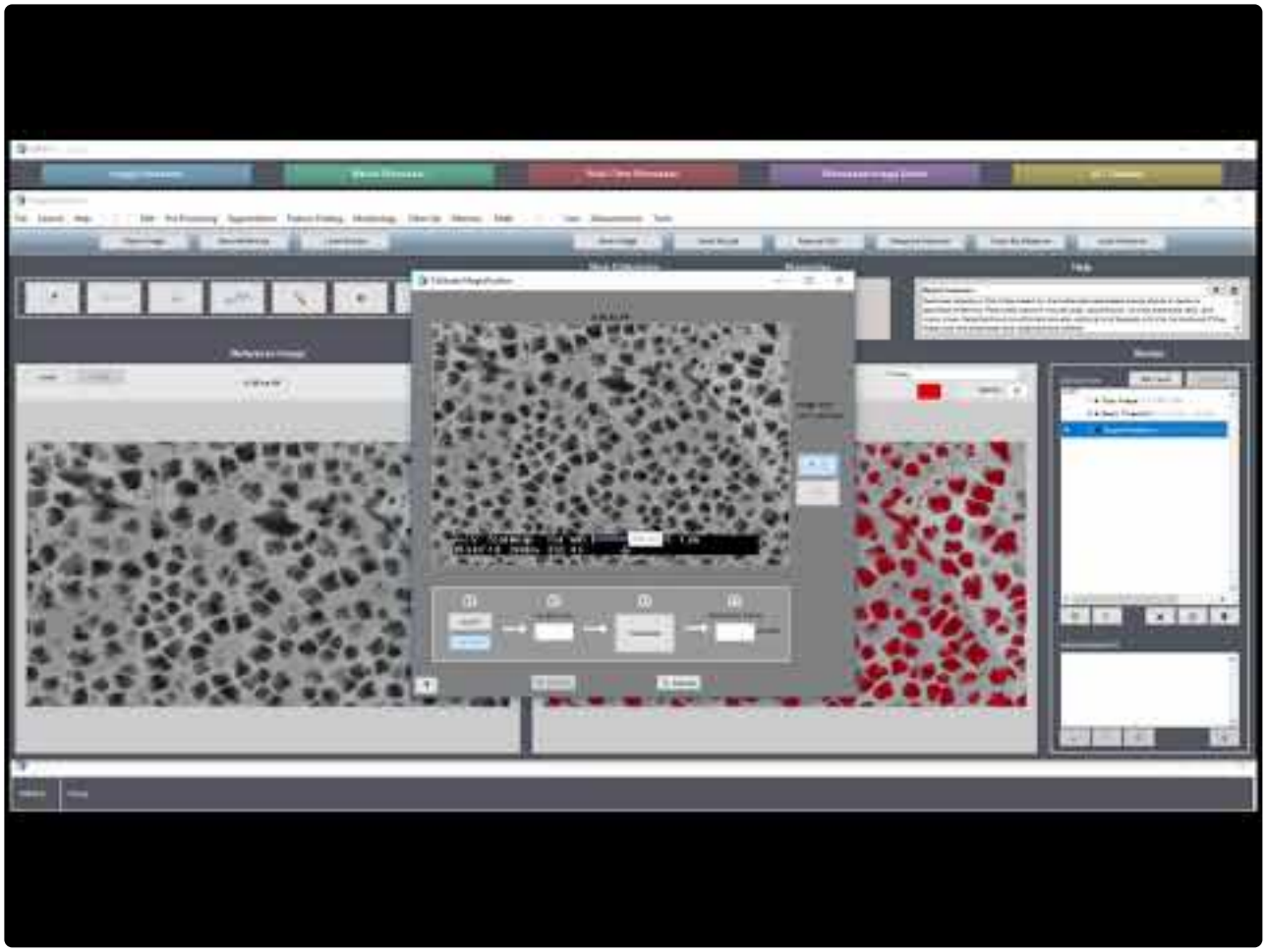


https://www.youtube.com/embed/GSrwRvSFh4E?rel=0

Polishing

Calibrating The Scale

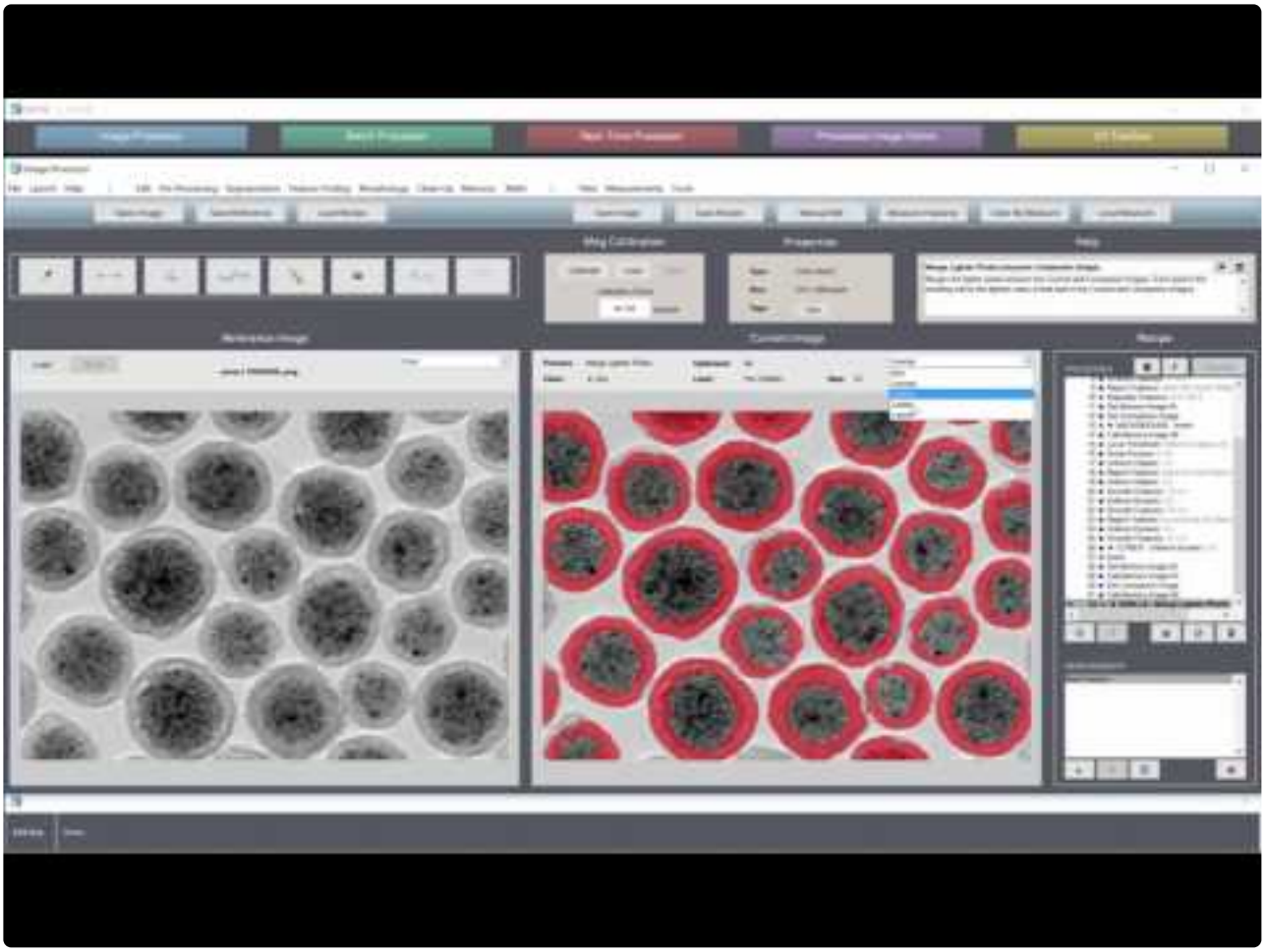
Shows how to calibrate the scale (i.e. calculate the pixel size) of your image, so that measurements can be made in physical units of microns, instead of pixels.



https://www.youtube.com/embed/il4U7UuiVbc?rel=0

Setting Layers

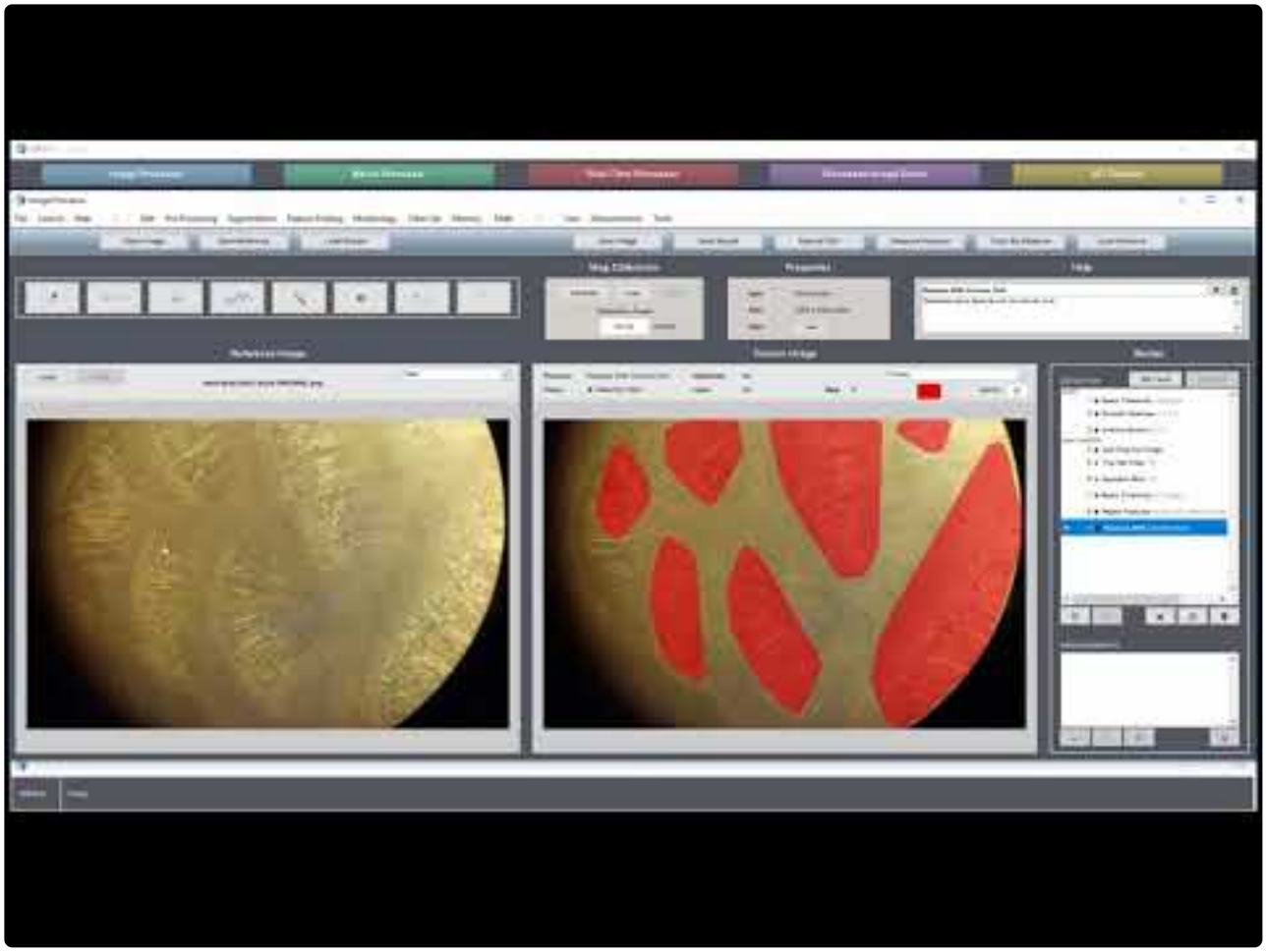
Discusses the power and procedures behind setting recipe steps as Layers.



https://www.youtube.com/embed/zbU_-hh6m4M?rel=0

Setting Chapters

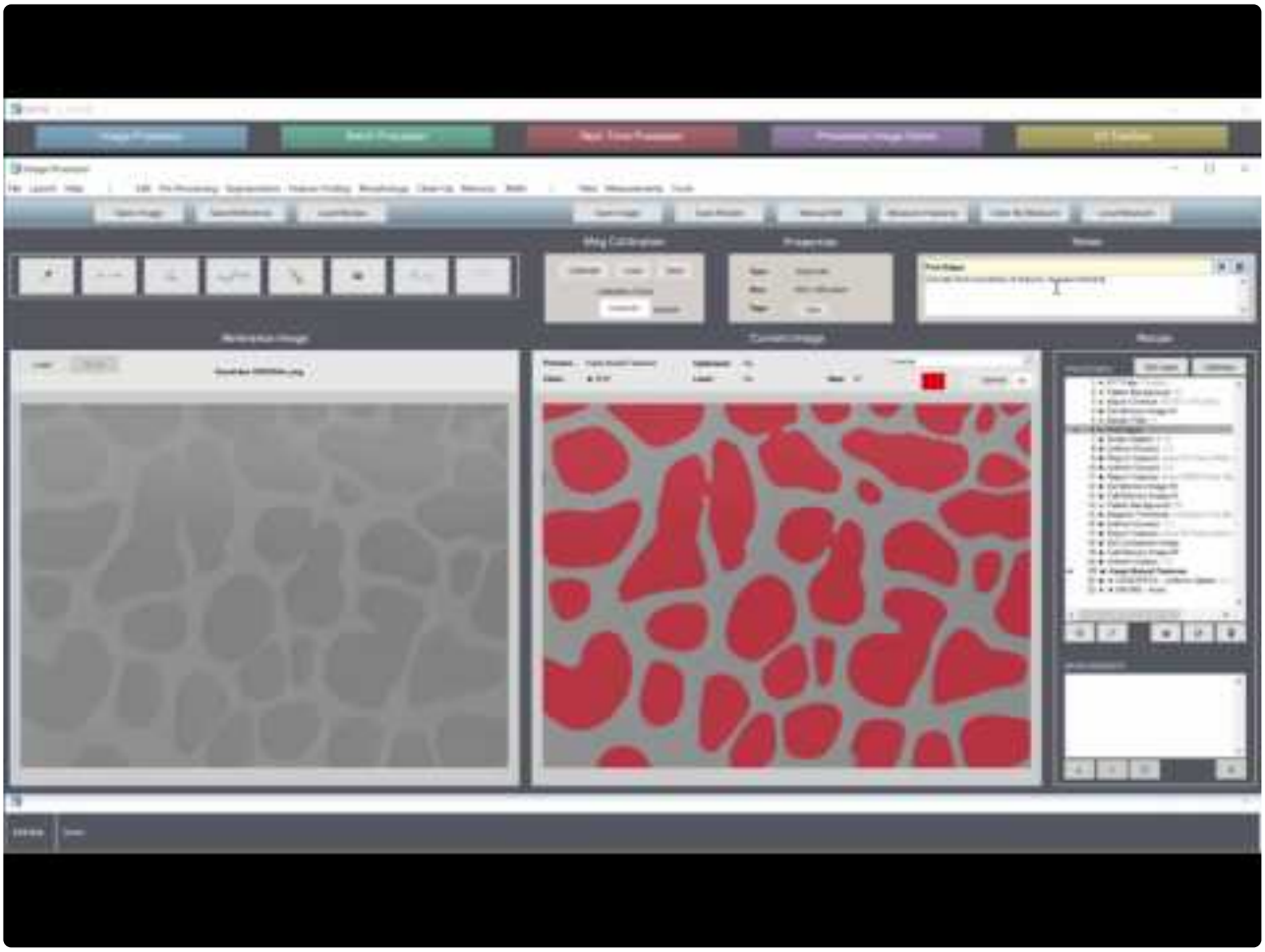
Discusses the advantages and procedures behind recipe Chapters.



https://www.youtube.com/embed/_h3zoYKzdTM?rel=0

Adding Notes & Flags

Demonstrates how to add Notes and Flags to Recipe steps.

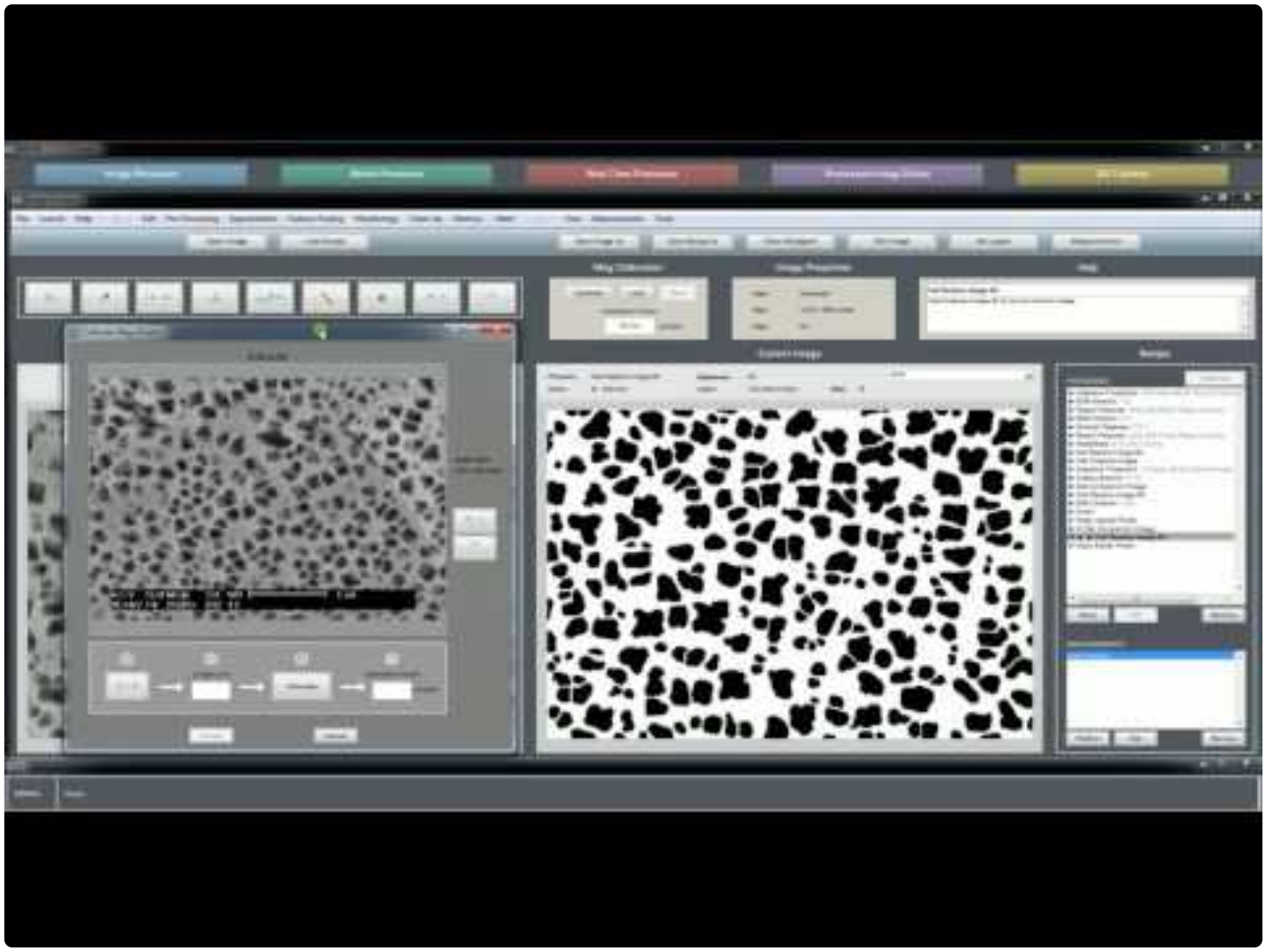


https://www.youtube.com/embed/xg55EgZbrUg?rel=0

Measuring

Generating Measurements

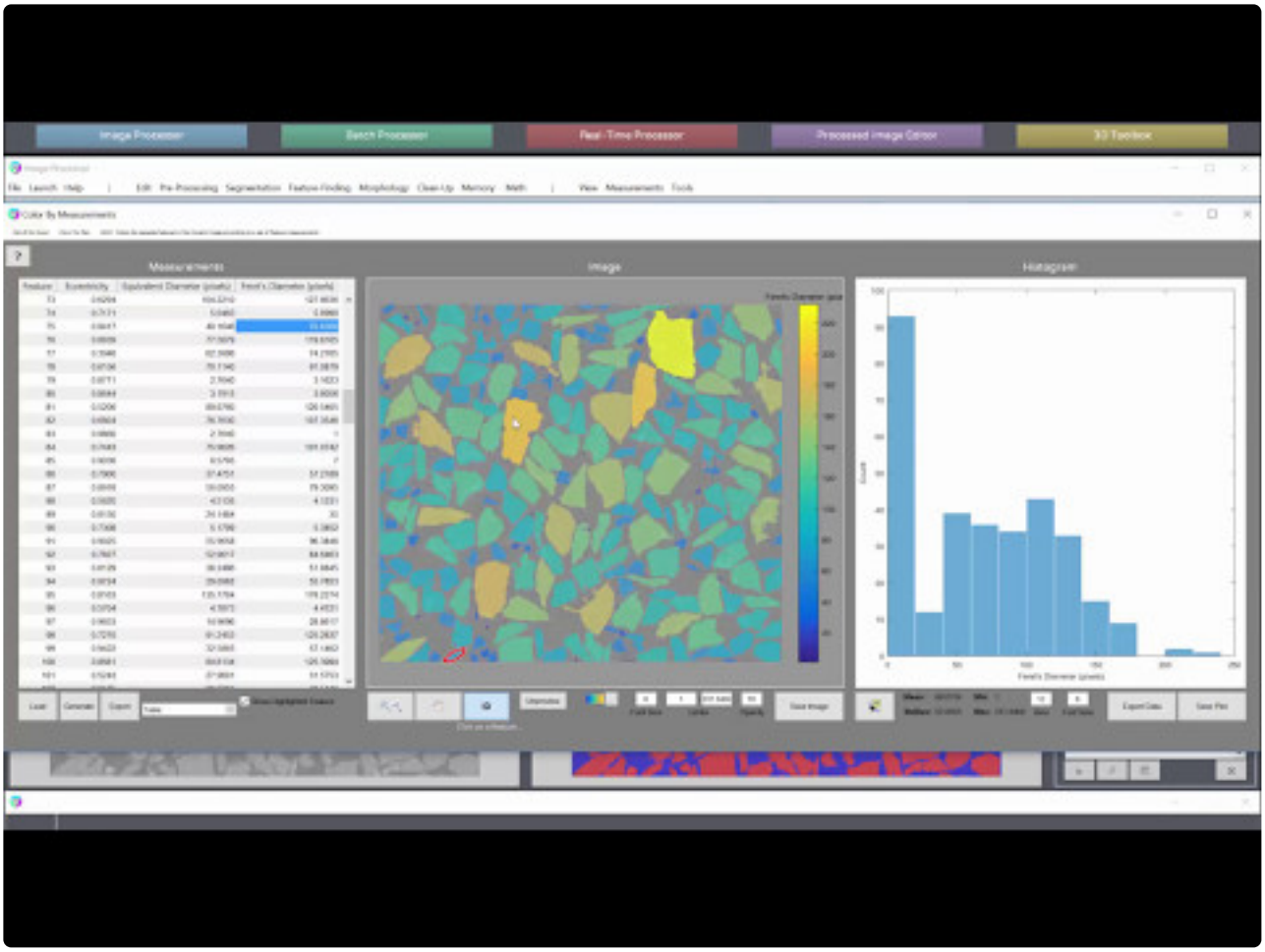
Demonstrates how to generate Global and Feature Measurements from segmented features of interest in an image.



https://www.youtube.com/embed/m_Nd_ETglH4?rel=0

Coloring By Measurements

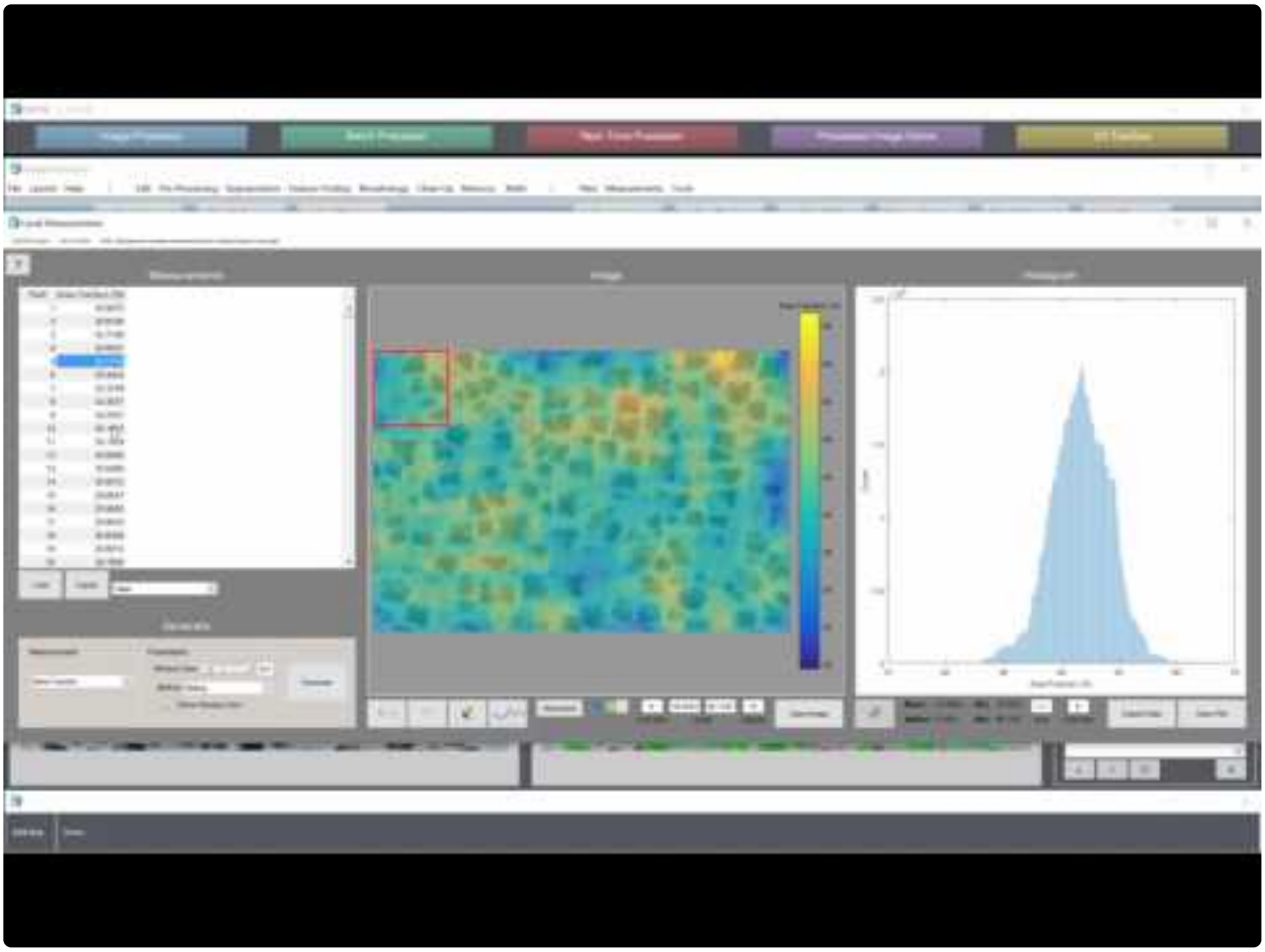
Demonstrates how to color segmented features according to any of their available measurements.



https://www.youtube.com/embed/kvhZrzKOubE?rel=0

Local Measurements

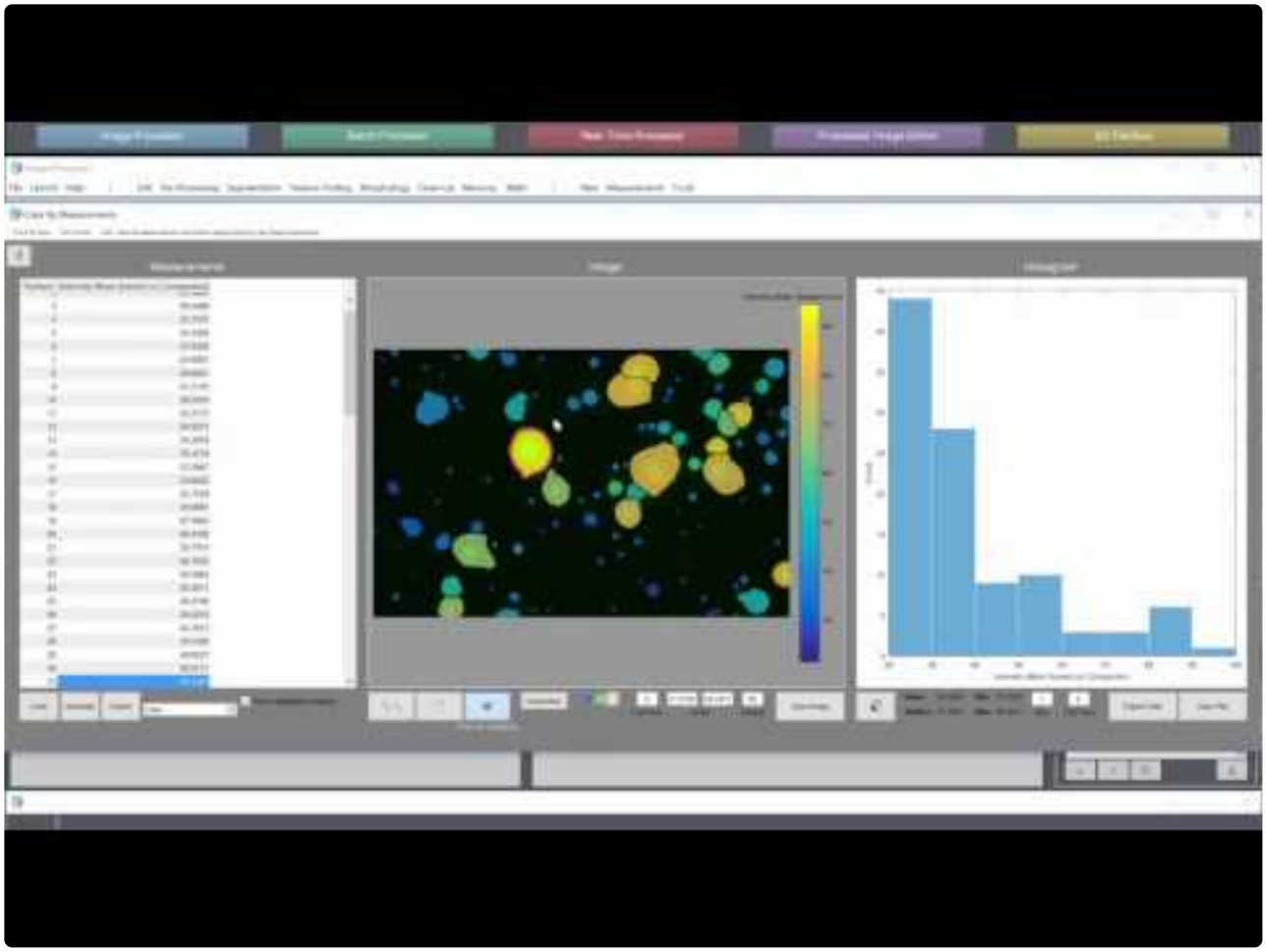
Demonstrates how to extract and visualize Local Measurements such as area fraction and thickness.



https://www.youtube.com/embed/WhAKluBliXs?rel=0

Measuring Feature Intensities

Demonstrates how to measure aspects of grayscale intensities within selected features.

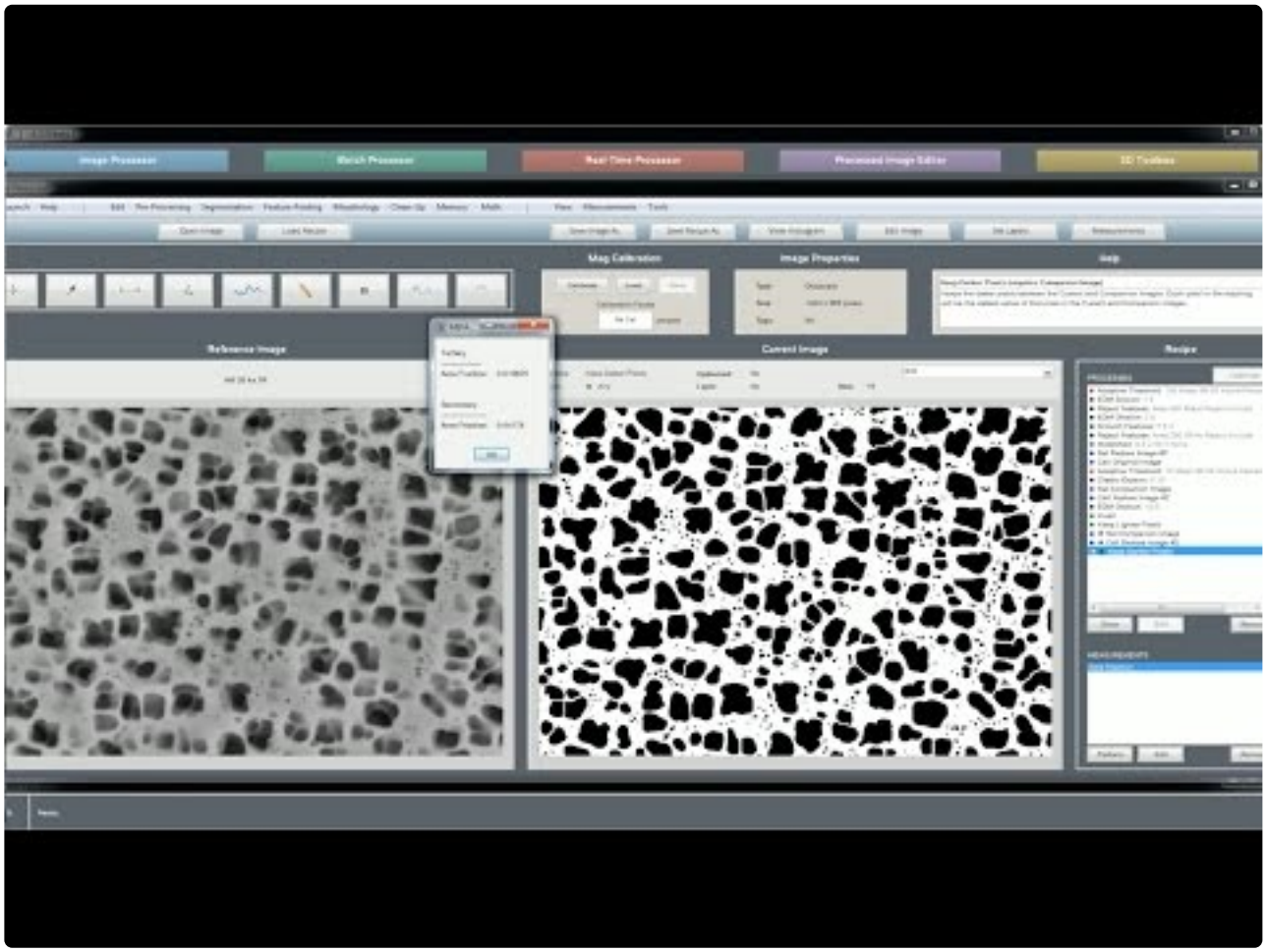


https://www.youtube.com/embed/DjX1gcxsKOM?rel=0

Batch Processor

Running a Batch

Shows how to setup and run a batch process to segment and measure features from a set images.

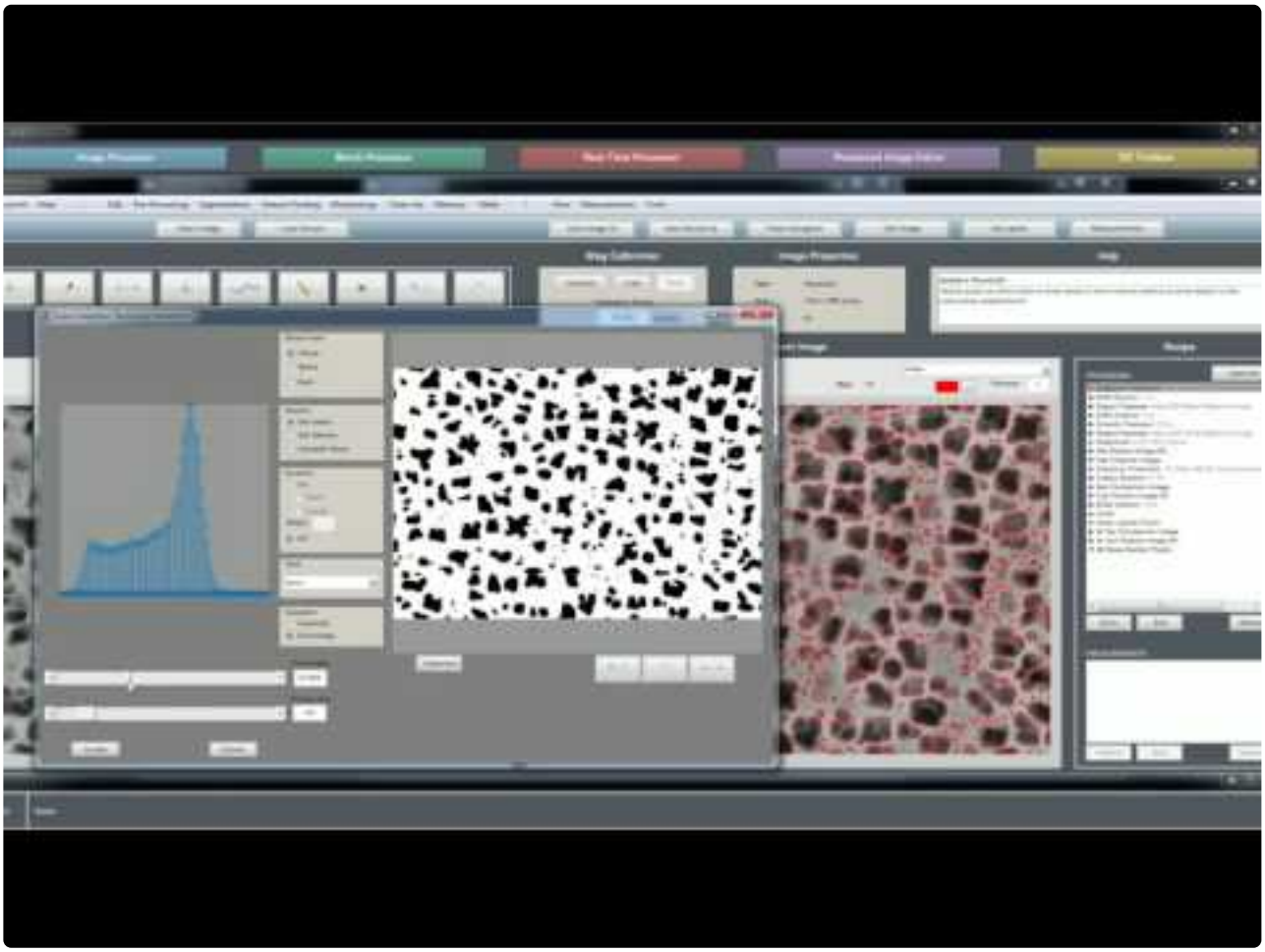


https://www.youtube.com/embed/KXbY8vPeroM?rel=0

Session Processor (formerly Post Processor)

Reviewing a 2D Batch

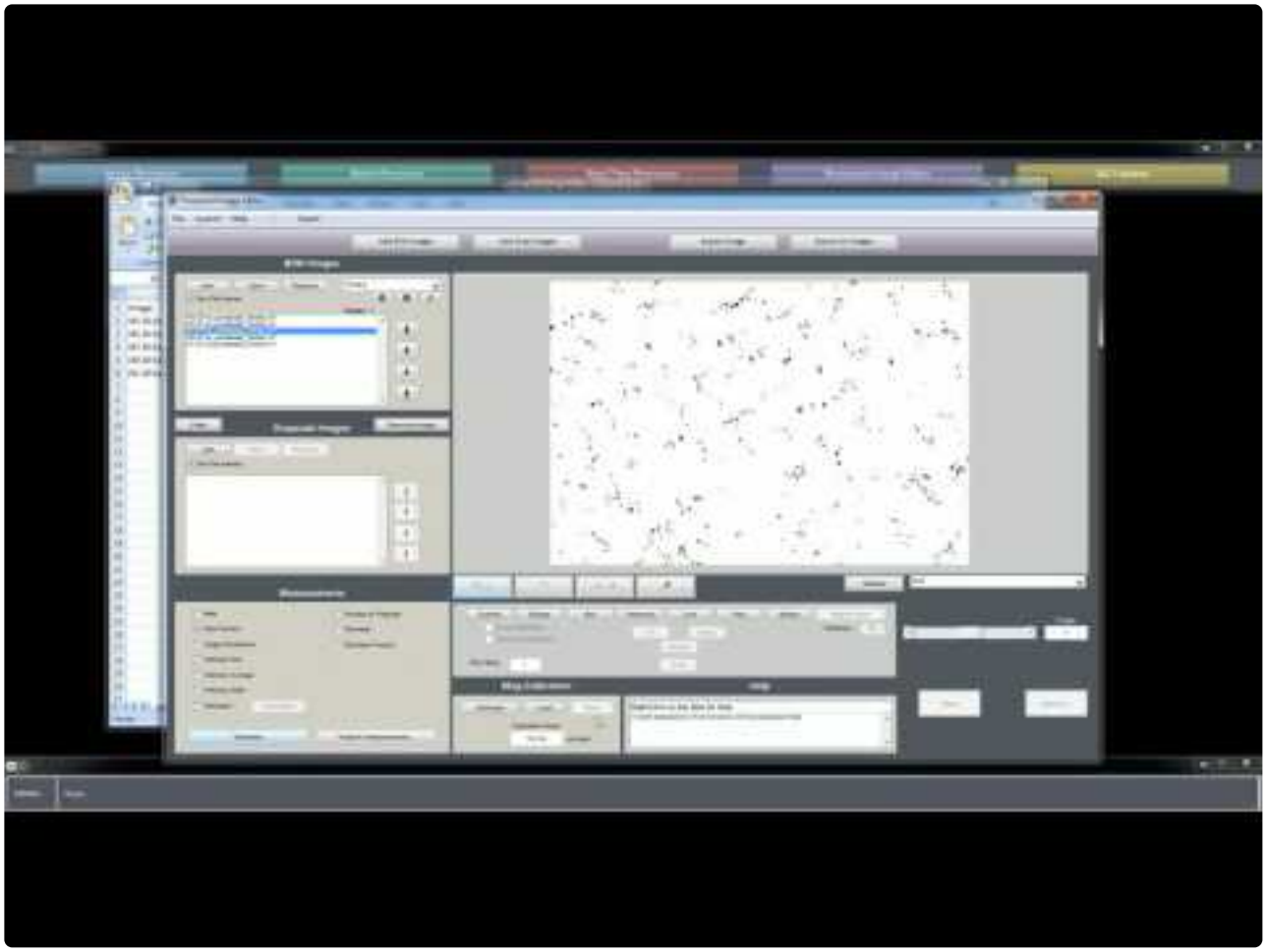
Shows how to review and edit multiple images after they have been batch processed to segment features of interest.



https://www.youtube.com/embed/whP1423oY1k?rel=0

Generating Measurements

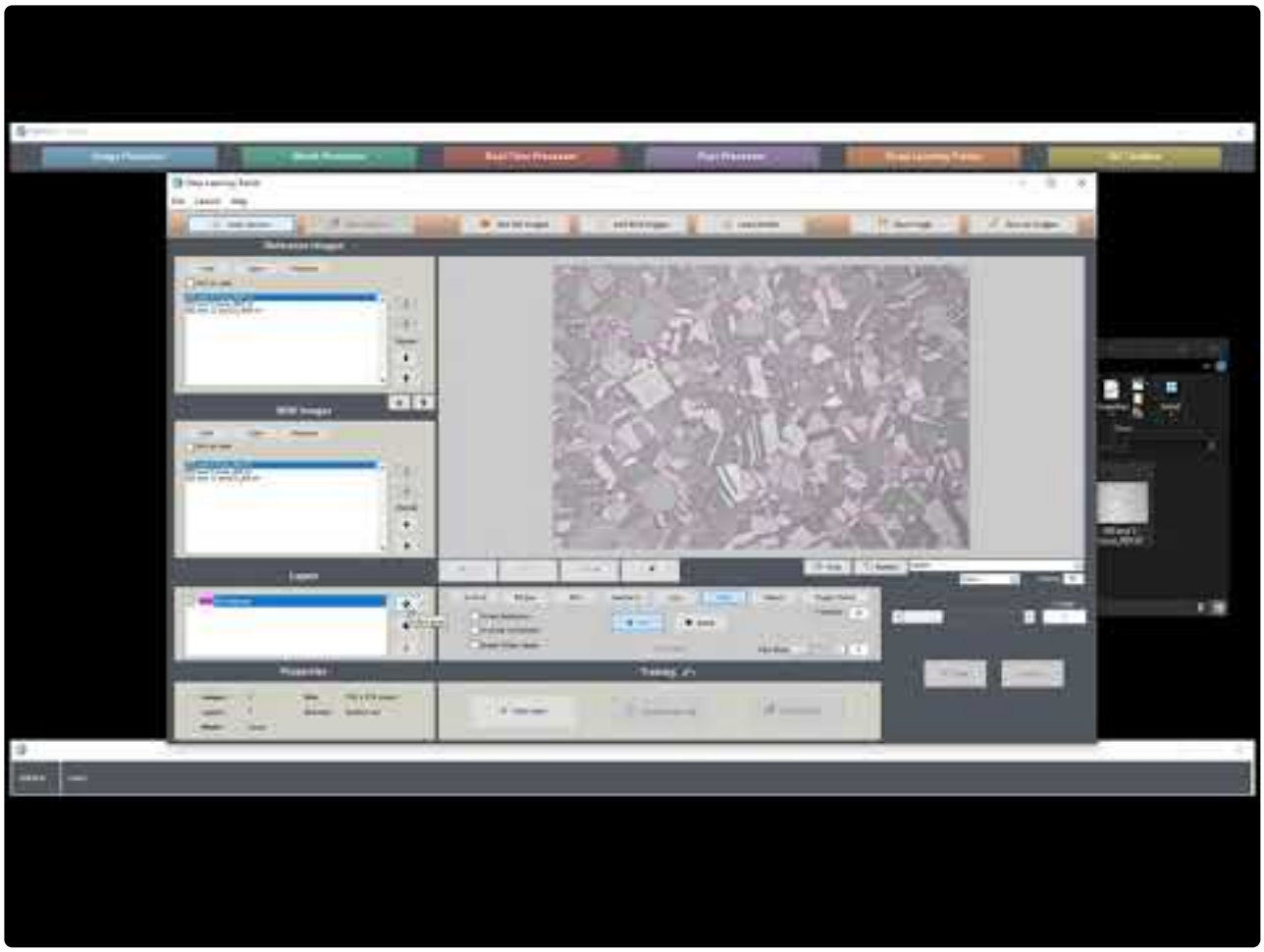
Shows how to generate measurements from multiple images after they have been batch processed to segment features of interest.



https://www.youtube.com/embed/KyzyMeVz0bM?rel=0

AI Session Processor (formerly Deep Learning Trainer)

Demo Video illustrating the use of the AI Session Processor



https://www.youtube.com/embed/ACvg6LAsqRY?rel=0

Tracing

Shows examples of tracing features of interest to develop training data.

Training

Shows example of training a deep learning model.

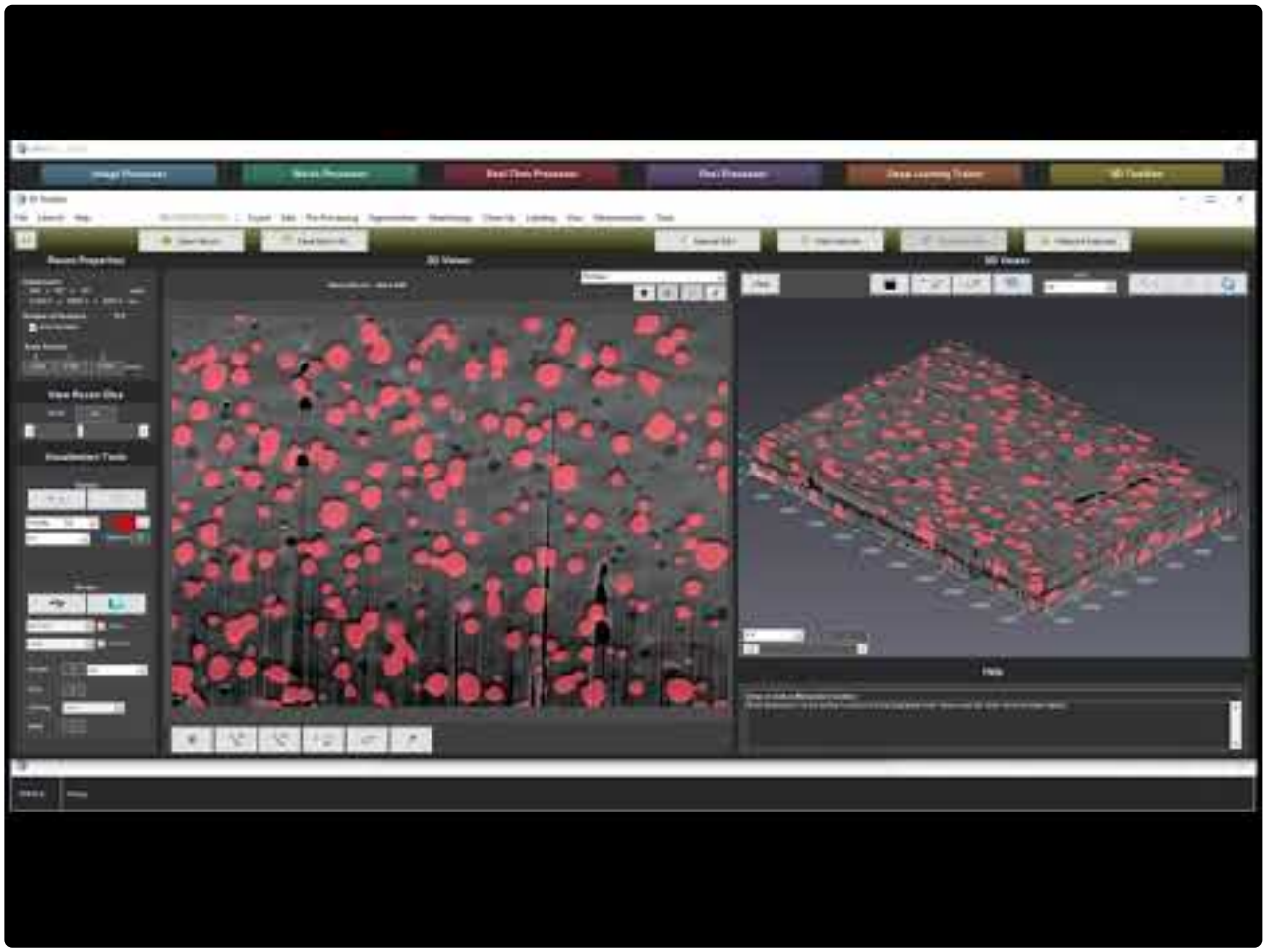
Applying

Shows example of applying a deep learning model to a new image.

3D Toolbox

Overview of 3D Reconstruction

Shows the workflow for creating, processing, visualizing, and measure a 3D Reconstruction.



https://www.youtube.com/embed/sTMwAB9xPgQ?rel=0

Downloads

You may download the below assets to use as learning tools and follow this tutorial.

Raw Dataset

Calibration Data

Segmentation Recipe

Recipe Store

Enter Recipe Store ››

What Is It?

The Recipe Store is a fantastic resource for getting the most out of MIPAR. Find and download Recipes for all sorts of segmentation problems. You'll learn fast and accelerate your work.

How Does It Work?

Browse the Recipes. Download the ones you like. We'll post examples next to them when possible. Every Recipe has common info so you know what you're getting.

Custom Recipes

One of MIPAR's unique features is the ability for you to receive custom Recipes from our experts anytime. Just submit an image here, and we'll return a Recipe for your task as quickly as possible. If needed, a member of our team will reach out for additional information.

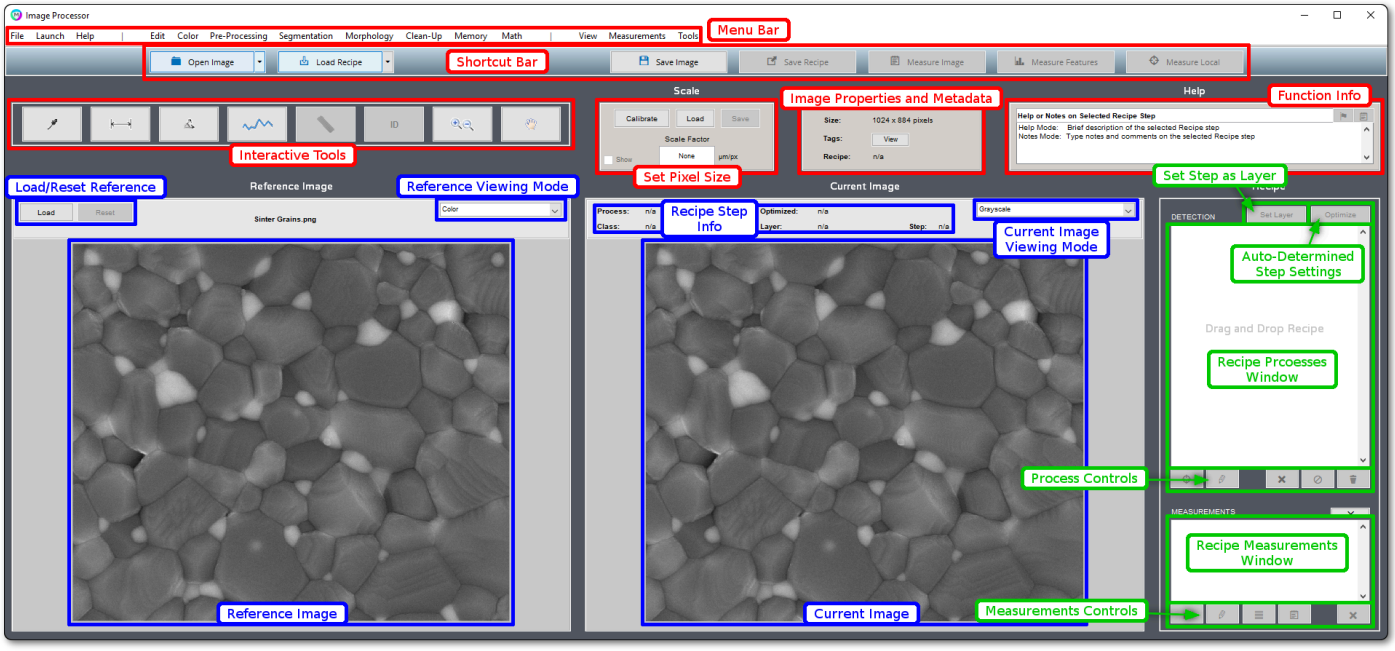
Image Processor

The Image Processor is the most commonly used app. It is used for building and editing Recipes, as well as applying Recipes to single images. Recipes are sequences of image processing steps, in the right order and with the right settings, which work to identify features of interest from your image.

Topics

Layout

Below is labeled screenshot of the Image Processor which reveals the layout of and purpose behind each user interface element.



See Keyboard Shortcuts for shortcuts relevant to the Image Processor.

The Recipe

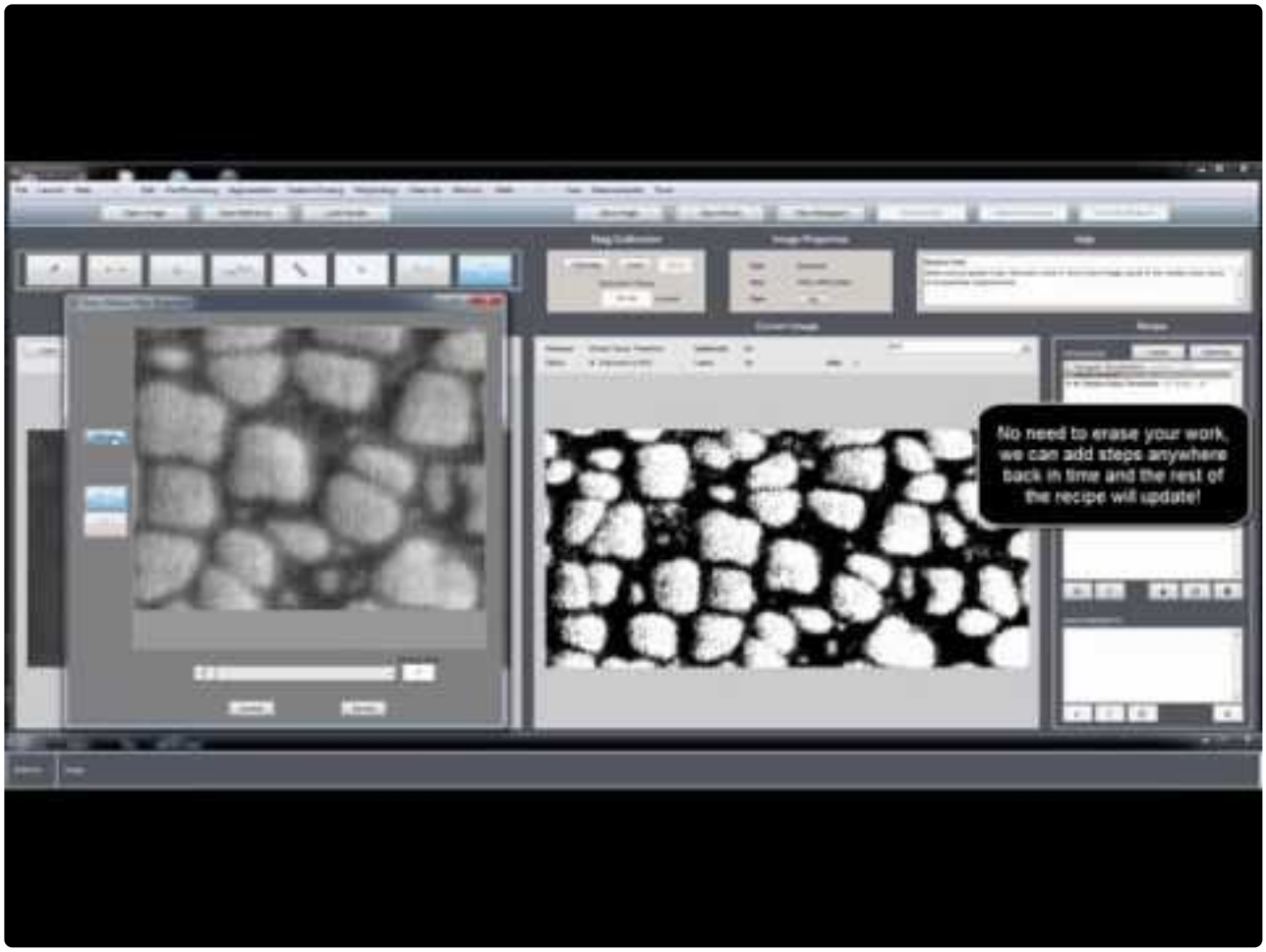
MIPAR's unique technology is the Recipe. Like an engine of a car, you use the one that's right for the job. Make them yourself, download from our free store, or get one custom-designed for you.

Recipes are built in the Image Processor, and are a sequence of image processing steps, in the right order and with the right settings, which work to identify features of interest from your image. Each step you take gets automatically recorded. Recipes are completely non-destructive, meaning that previous steps can be edited or removed, new steps inserted, and changes will propagate down the rest of the sequence. Any step of a Recipe can be double-clicked to show its result.

In addition to processing steps, Recipes can also contain global measurements. If there are no Layers in a Recipe when it is run on an image or a batch, all global measurements will be performed on the last Recipe step. If there are layers, all global measurements will be performed on each layer.

Once saved, a Recipe can be applied to another image in the Image Processor, or automatically to a set of images in the Batch Processor. You'll be amazed at the range of problems you can solve with this technology.

Demo

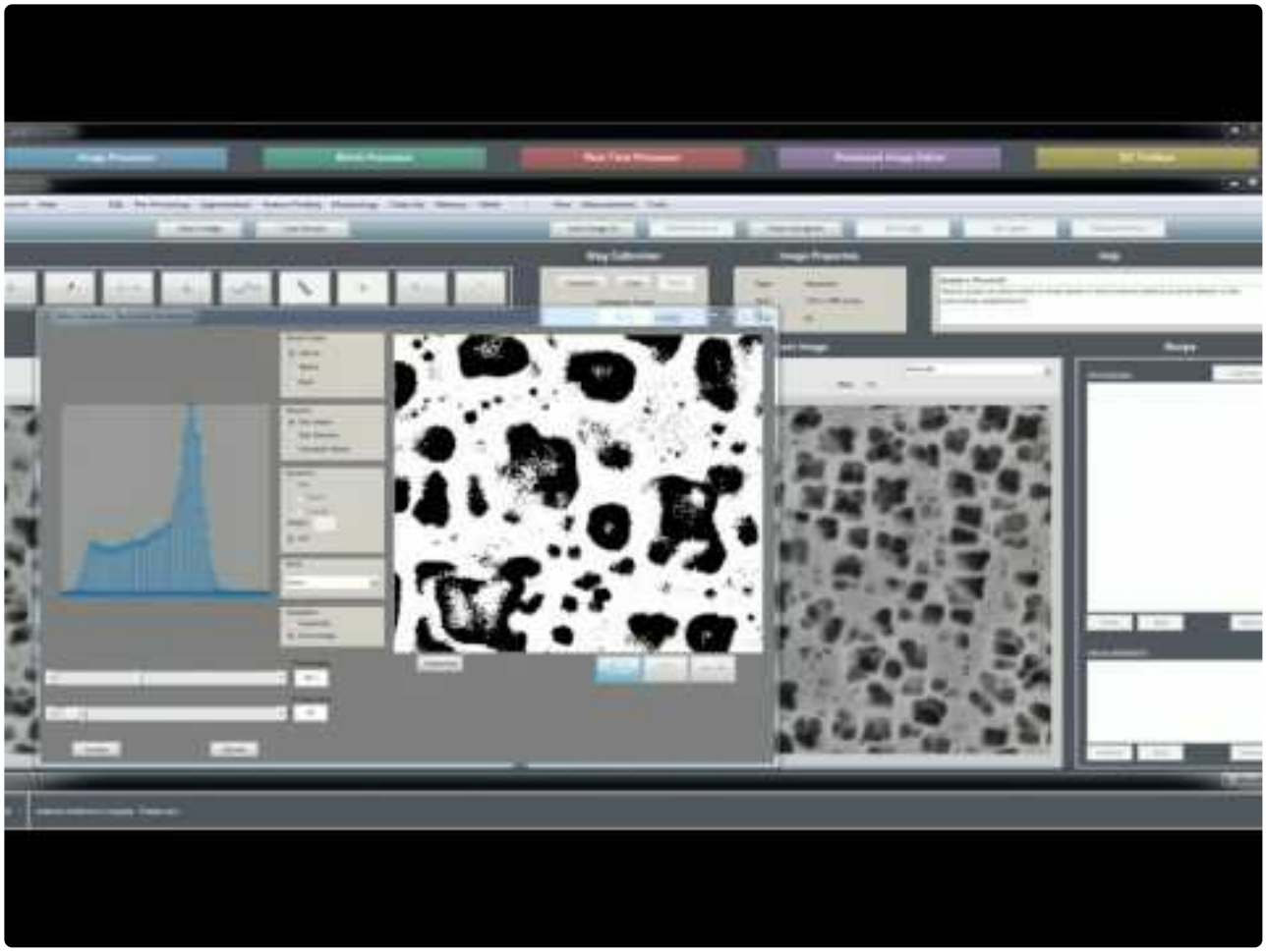


https://www.youtube.com/embed/Qh7JxzDWwUU?rel=0

Custom Recipes

One of MIPAR's many features that no competitor can match is the ability to receive custom recipes from our expert anytime. Just submit an image here, and we'll return a Recipe for your task as quickly as possible.

Tutorial



https://www.youtube.com/embed/D01HYV8yAiw?rel=0

https://www.youtube.com/embed/D01HYV8yAiw?rel=0

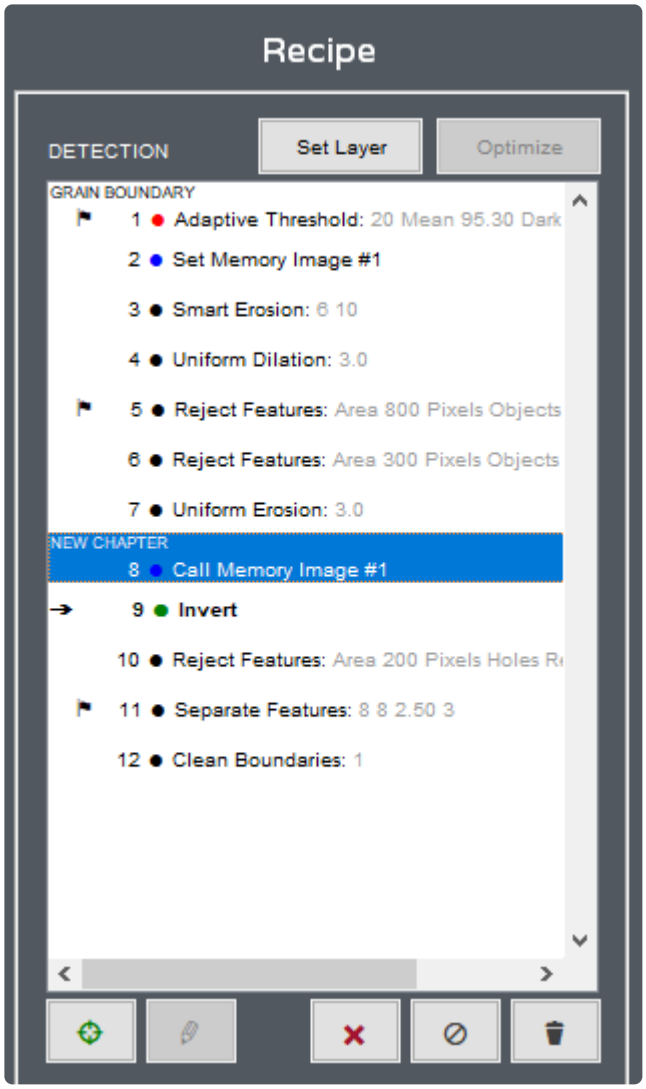
Chapters

Chapters allow a Recipe to be broken up into sections, where the first step in each section (Chapter) is given a name which generally describes the purpose of that series of steps.

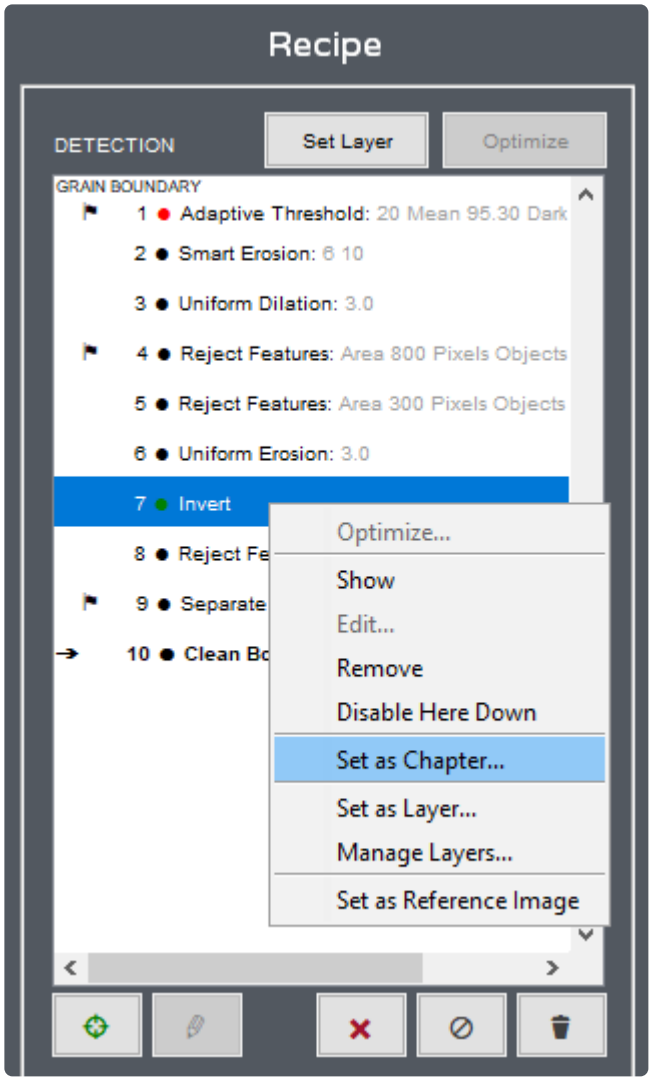
Setting Chapters

During the building of a Recipe, the addition of certain steps will automatically trigger the start of a new Chapter, since these steps typically indicate the start of new overall goal or purpose in the Recipe. The steps which automatically trigger the start of a new Chapter are:

- Call Memory Image #1-6
- Call Companion Image
- Load Companion Image
- Call Original Image
- Channel Operation
- Color Select
- Blank



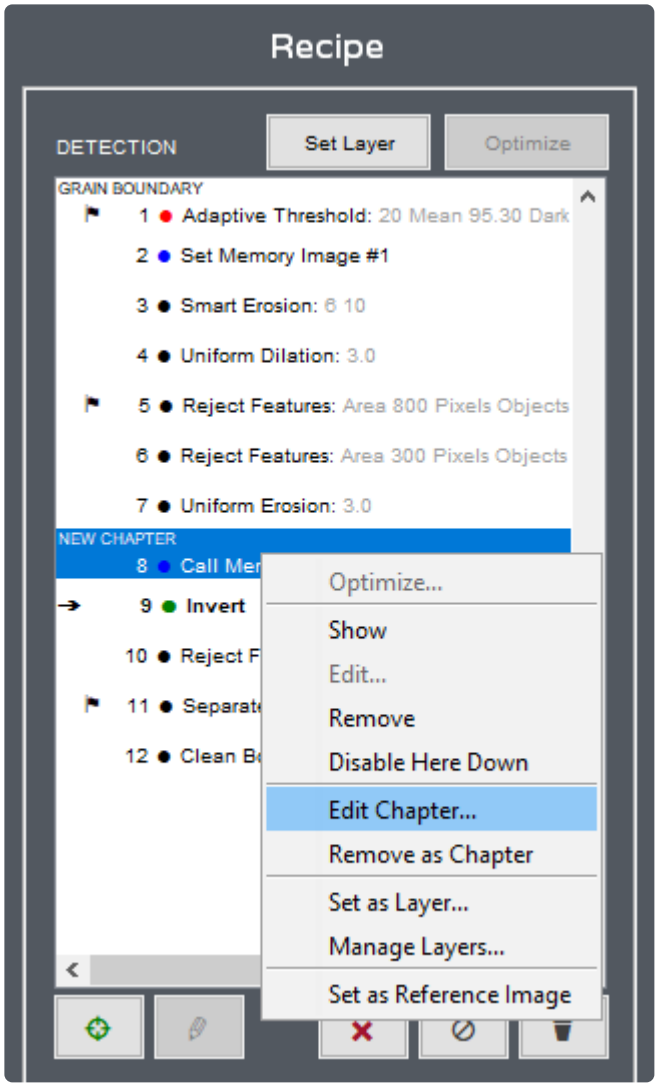
Any Recipe step can be manually set as the start of a new Chapter at anytime. To do so, right-click on the step, choose “Set as Chapter…”, type the desired name, and click “OK”.



Editing Chapters

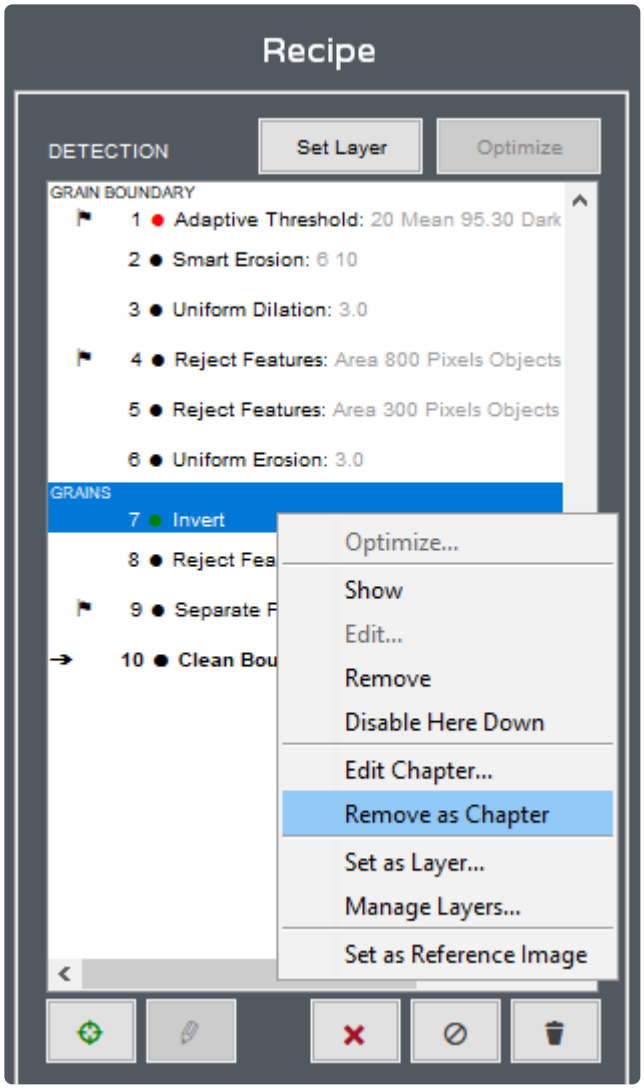
The name of any Chapter can easily be changed at anytime. To do so, right-click on the step, choose “Edit Chapter…”, type the desired name, and click “OK”.

Chapter names are set automatically when a step within the Chapter is set as a Layer, but only if the current name of the Chapter the is the default “New Chapter” (or “Start” in the case of the first step). If this is the case, the Chapter will automatically be given the same name as the name of the Layer. If the Chapter had already been named something other than the default, the Chapter name will not be changed.

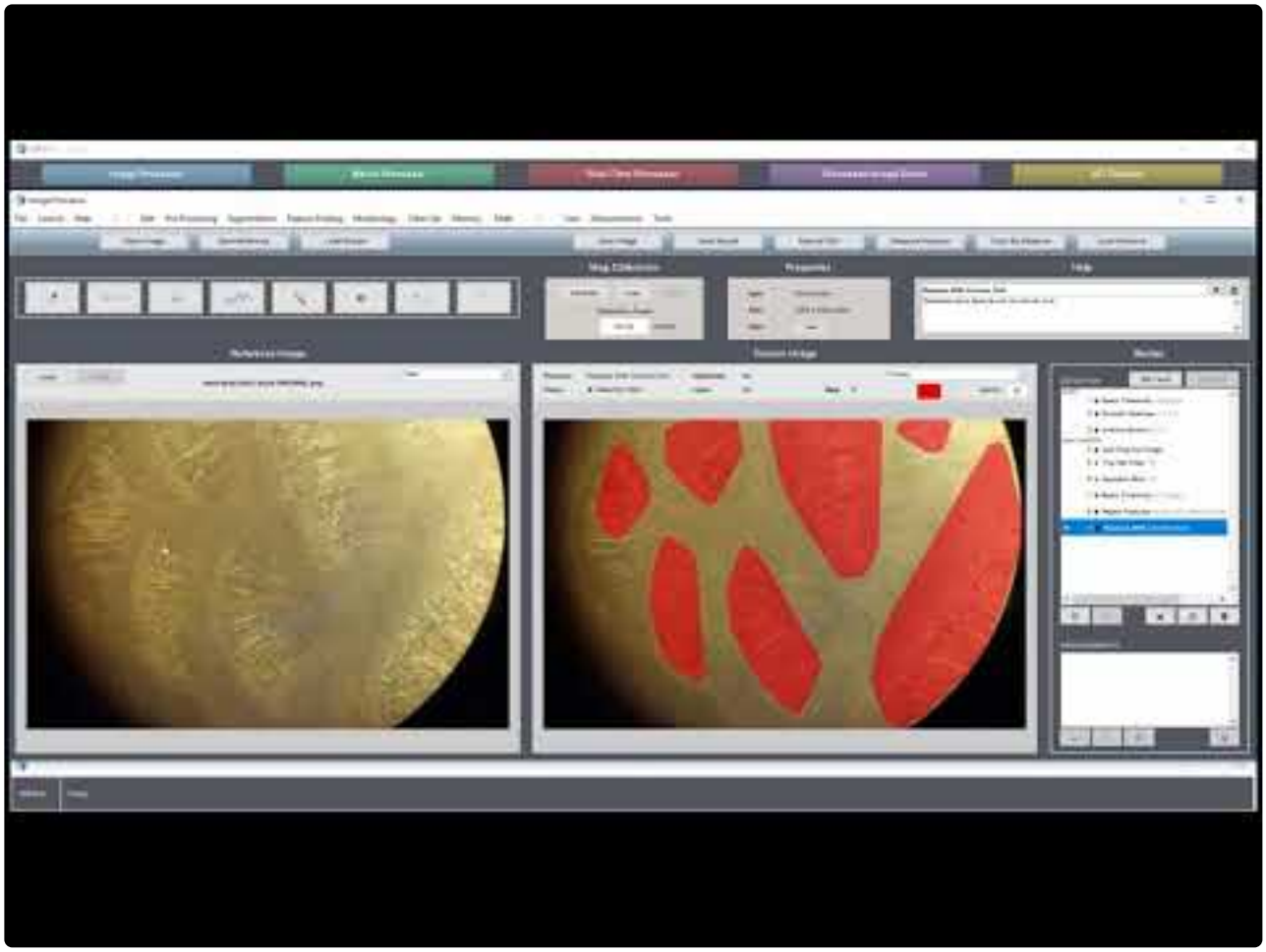


Deleting Chapters

Any step (except for the first) can be un-set as a Chapter at anytime. To do so, right-click on the step and choose “Remove as Chapter”.


Recipe
DETECTION
Set Layer
Optimize
GRAIN BOUNDARY
1 Adaptive Threshold: 20 Mean 95.30 Dark
2 Smart Erosion: 6 10
3 Uniform Dilation: 3.0
4 Reject Features: Area 800 Pixels Objects
5 Reject Features: Area 300 Pixels Objects
6 Uniform Erosion: 3.0
GRAINS
7 Invert
8 Reject Fea
9 Separate F
10 Clean Bou
Optimize...
Show
Edit...
Remove
Disable Here Down
Edit Chapter...
Remove as Chapter
Set as Layer...
Manage Layers...
Set as Reference Image

Tutorial



https://www.youtube.com/embed/_h3zoYKzdTM?rel=0

https://www.youtube.com/embed/_h3zoYKzdTM?rel=0

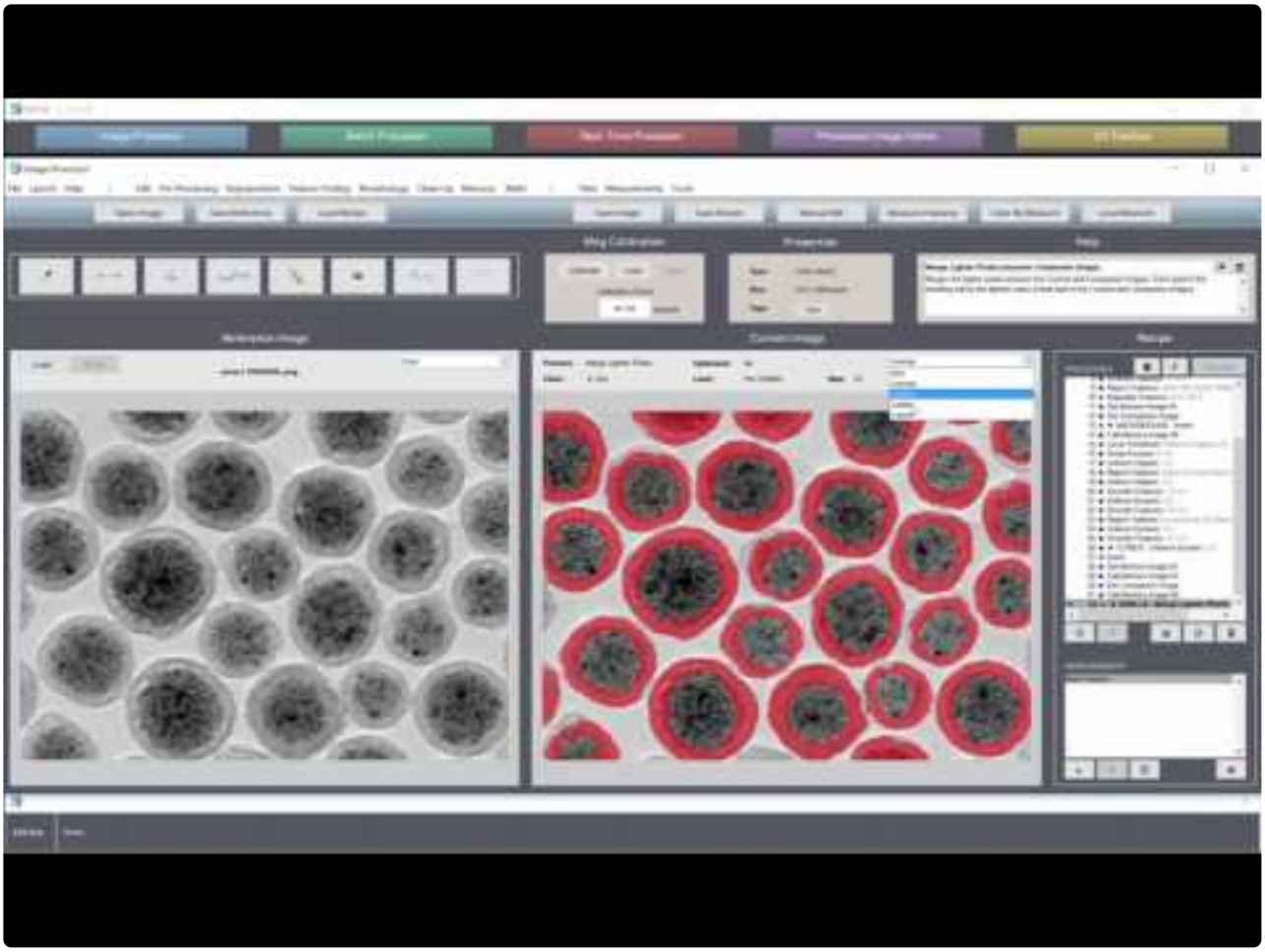
Layers

Each Recipe step outputs either an grayscale, or B/W image. B/W images are often the goal, as the **black pixels represent the selection** at that point in the Recipe. If you are only targeting one set of features, a final B/W image is all you need. However, if you need to identify multiple feature types (e.g., large vs. small particles), Layers come in very handy.

Layers are essentially “favorited” Recipe steps, which can be overlaid together and measured separately. For example, if you’re looking for large vs. small particles, you may have one step which selects the large, and one which selects the small. Just set each step as a Layer, and you’re done. You’ve now separately classified your two feature types and can enjoy coloring them separately, outputting each selection during a batch process, and taking measurements from each with one click.

To move a Layer from one step to another, select then Layer step, hold CTRL and select the step to move to. The “Set Layer” button will change to “Move Layer”. Click this button to move the layer.

Tutorial



https://www.youtube.com/embed/zbU_-hh6m4M?rel=0

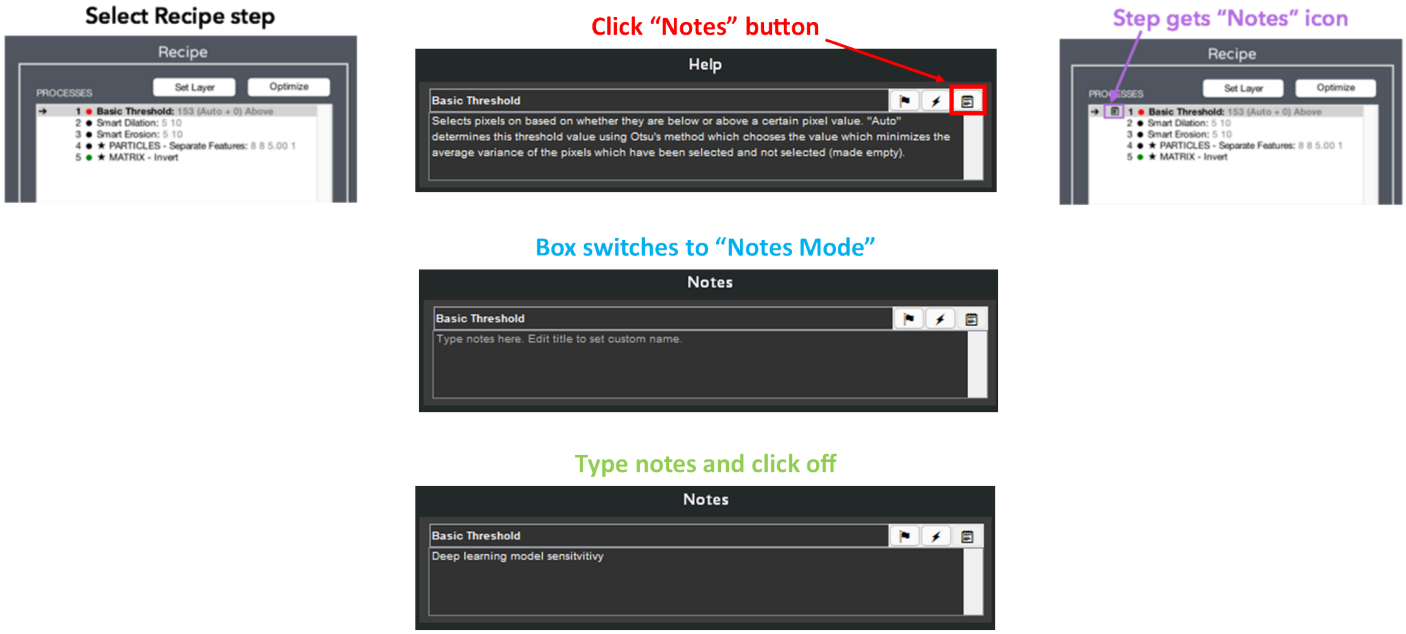
https://www.youtube.com/embed/zbU_-hh6m4M?rel=0

Notes, Flags, Interruptible, Custom Recipe Names

Notes

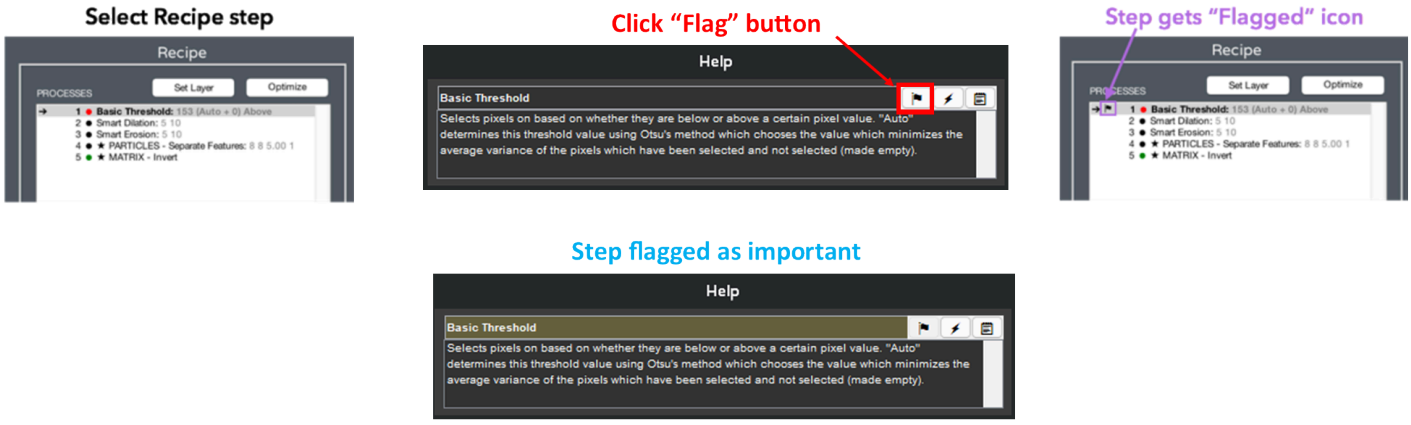
Notes are comments that users may add to any Recipe step. These can be useful for reminding yourself, or indicating to others, the intent behind a step. They can also be helpful in indicating which settings should be adjusted in situations when the Recipe needs tweaking.

Notes may be added to a step by selecting the step and clicking the “Notes” button in the “Help” box. Erasing the text from the “Notes” box will remove the notes from the step.



Flags

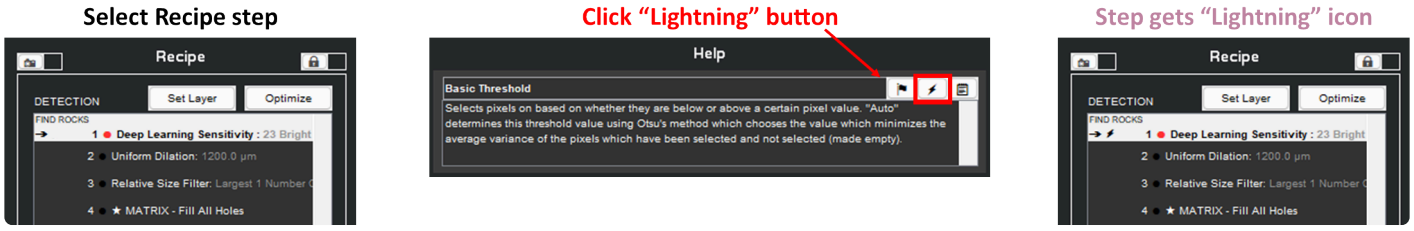
Flagging Recipe steps indicates that they are critical to the accuracy of the feature detection. Flagged steps are likely the first steps to need adjustment if features are not detected accurately when the Recipe is applied to a given image.



A step may be flagged by selecting the step and clicking the “Flag” button in the “Help” box. Un-click the “Flag” button to un-flag a step.

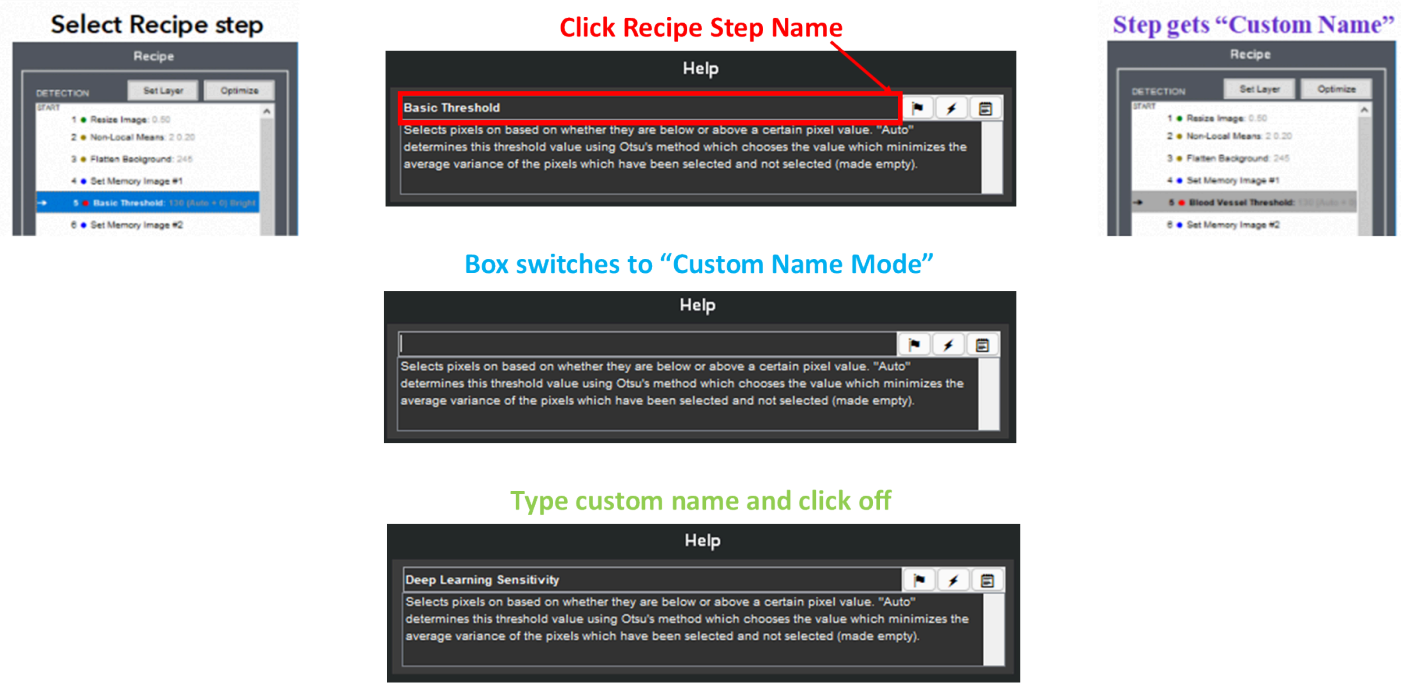
Interruptible

User editable steps can be set to interrupt recipe execution in the Image Processor, the Batch Processor or the Real Time Processor to ask for user input. To set the step as interruptible, select the recipe step and click the ‘Lightning’ button.

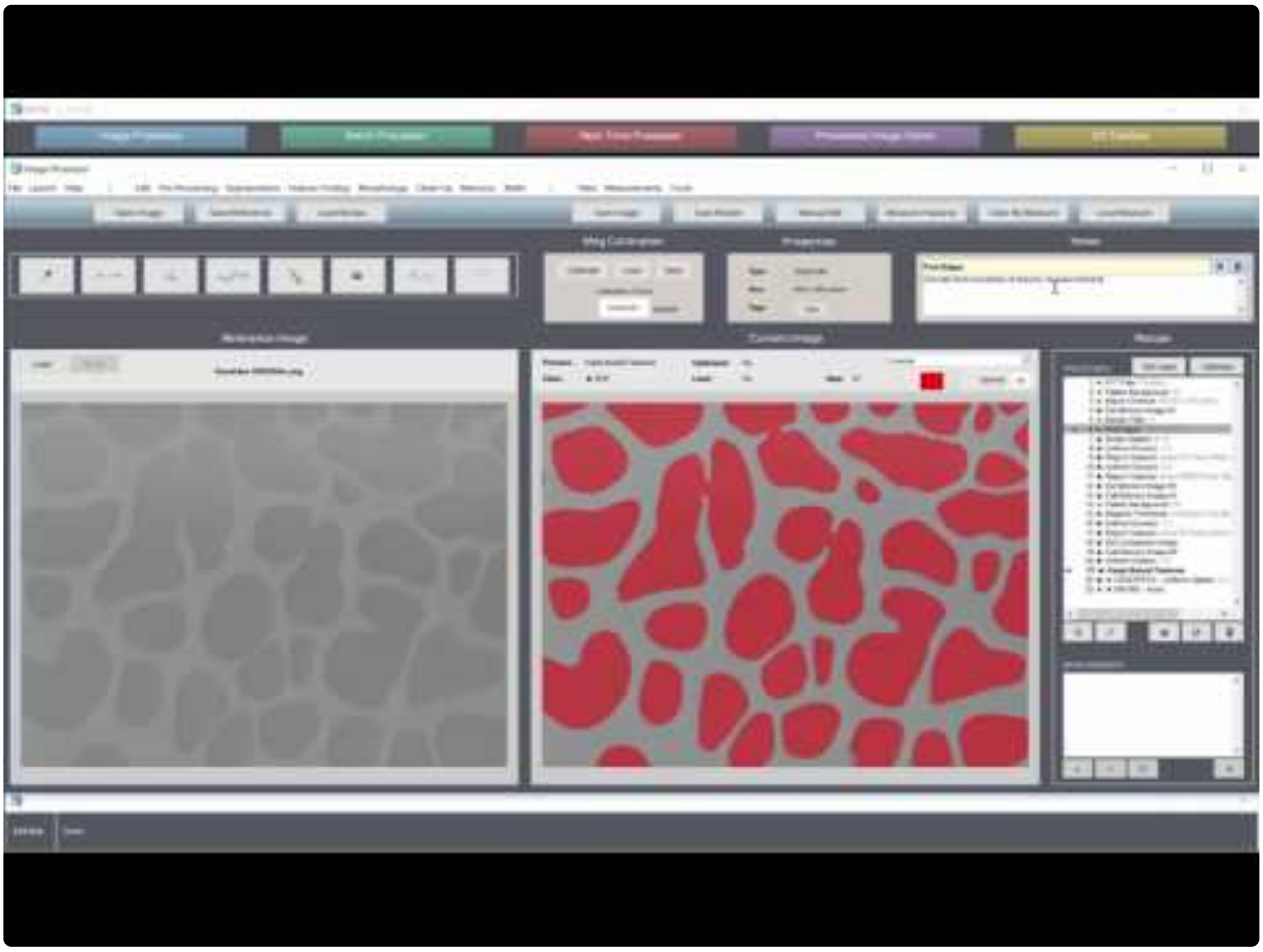


Custom Recipe Names

Custom recipe names are used in combination with flags and notes to annotate the recipe and help organize the workflow for users



Tutorial

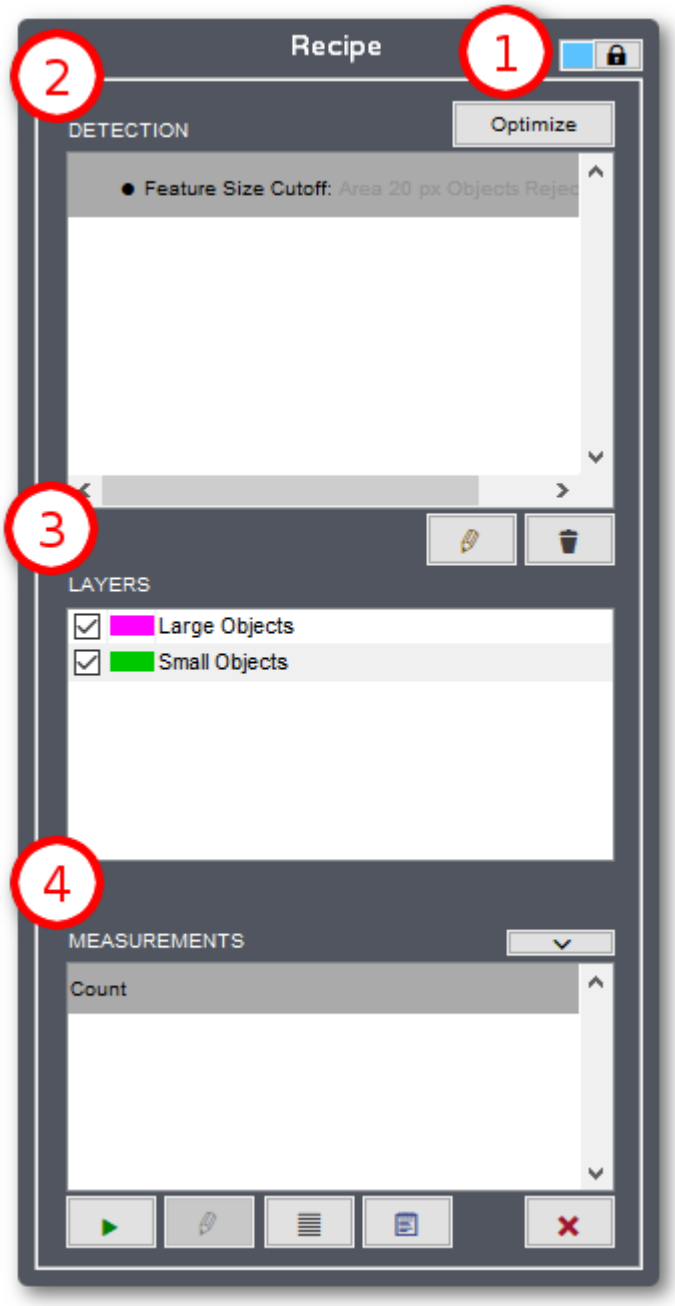


https://www.youtube.com/embed/xg55EgZbrUg?rel=0

https://www.youtube.com/embed/xg55EgZbrUg?rel=0

Simple Recipe Mode

Simple Recipe Mode collapses the Recipe display, showing only flagged steps and layers. This enables the Recipe end-user to easily ignore most steps and make adjustments only to important ones, without having to search through the entire Recipe. In addition, the Layers panel allows users to view and edit Layers with a much more streamlined feel.



1a. Switching Recipe Mode

To enable Simple Recipe Mode, the Recipe must have at least one layer. If it does, clicking the Lock icon

will toggle between Simple and Detail Recipe Mode.

1b. Password Protection

Password protection keeps a Recipe from being switched from Simple to Detail Mode and prevents measurements from being added or removed. Flagged steps exposed in Simple Mode will still be editable.

To set a password, right-click on the lock icon and select “Lock with Password”.

To open a locked Recipe, either left-click or right-click on the lock icon and enter the password when prompted. The Recipe Mode can then be switched back and forth without a password until the Recipe is re-loaded.

To change the password or remove the need for a password completely, right-click on the lock icon of an unlocked recipe and select “Edit Password …” or “Clear Password”.

2. Detection

The Detection panel shows flagged steps. These steps can be edited or optimized (if applicable) by either double-clicking, right-clicking and selecting “Edit…” or “Optimize…”, or selecting the step and using the Optimize or Edit buttons.

3. Layers

The Layers panel shows all layers in the Recipe . Specific layers can be shown or hidden from the Current Image pane by toggling the check box by each layer. Each layer’s color and name can also be changed in this panel by clicking on them.

4. Measurements

The Measurements panel displays all the measurements currently present in the Recipe and allows measurements to be added, modified, and run.

Calibrating The Scale

Embedded Image Scale Factors

MIPAR can read embedded scale factors from over 150 file formats once the Bio-Formats plugin is in place. See the Supported Formats page for more information on setting up this plugin.

If an image is opened which contains an embedded scale factor, the factor will be set in light-gray text in the “Scale” panel, and the corresponding calibration units will be set automatically.

Loading a Recipe with a scale factor, or setting a scale factor, will overwrite the image scale factor. Erasing the Recipe scale factor will revert back to the image scale factor.

Image scale factors are supported in the Image Processor, Batch Processor, Real-Time Processor, and Post Processor.

> ! **Automatic Scale Adjustment:** In the Image Processor, when adding or removing a *Resize Step* from a recipe, the scale will automatically adjust to compensate. In the Image, Batch, and Real-Time Processors, when loading an image where its scale is automatically read from its metadata, this scale will automatically be adjusted by the presence of Resize steps in the recipe. These adjustments apply to the entire recipe, even steps before the Resize step.

Calibrate Scale Tool

The Calibrate Scale tool allows users to calculate their pixel size, such that distance and areal measurements can be made in physical units, rather than pixels. The image used for calibration should contain an object of a known length (typically a scale bar), and should be of the same magnification and pixel density as the image to measure.

This tool can be used in 1 of 2 modes: Auto or Manual:

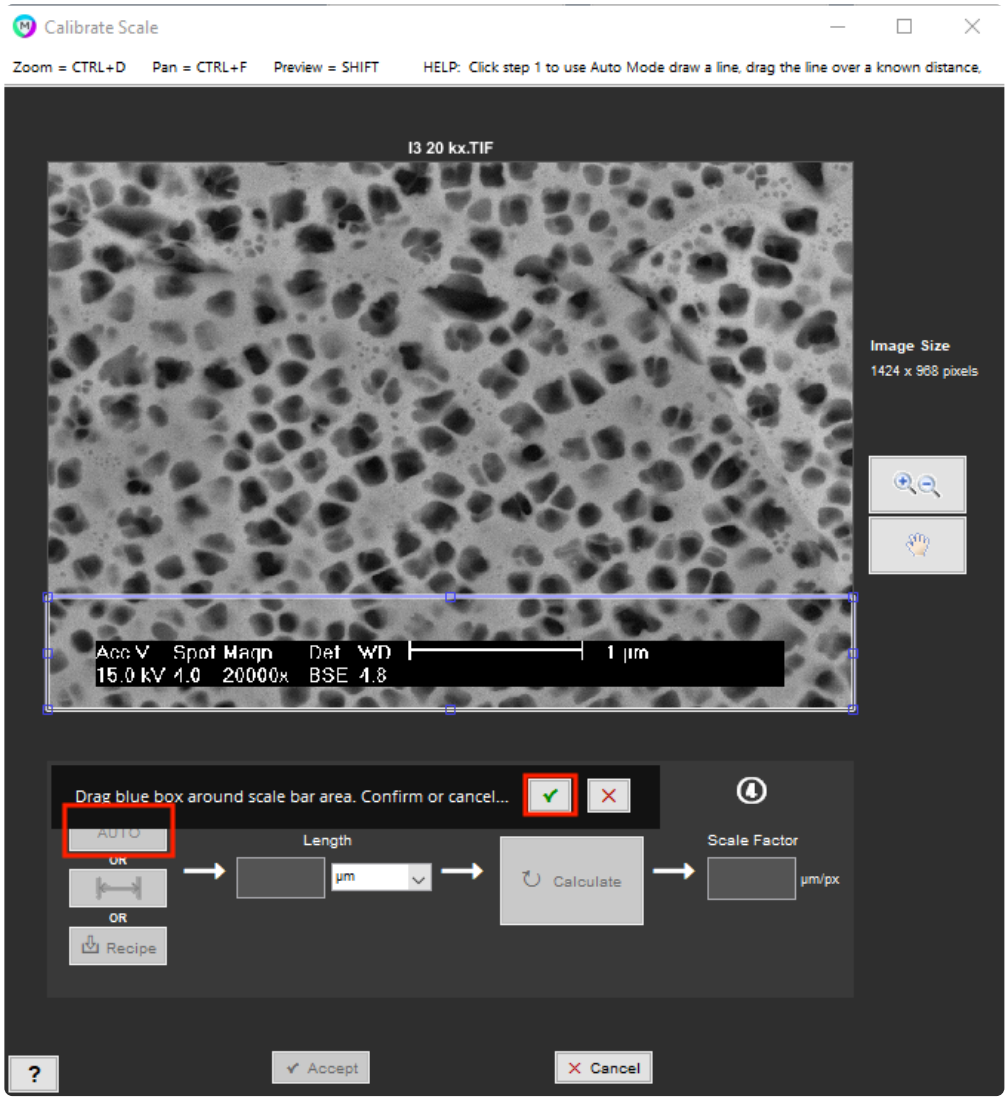
Auto Mode

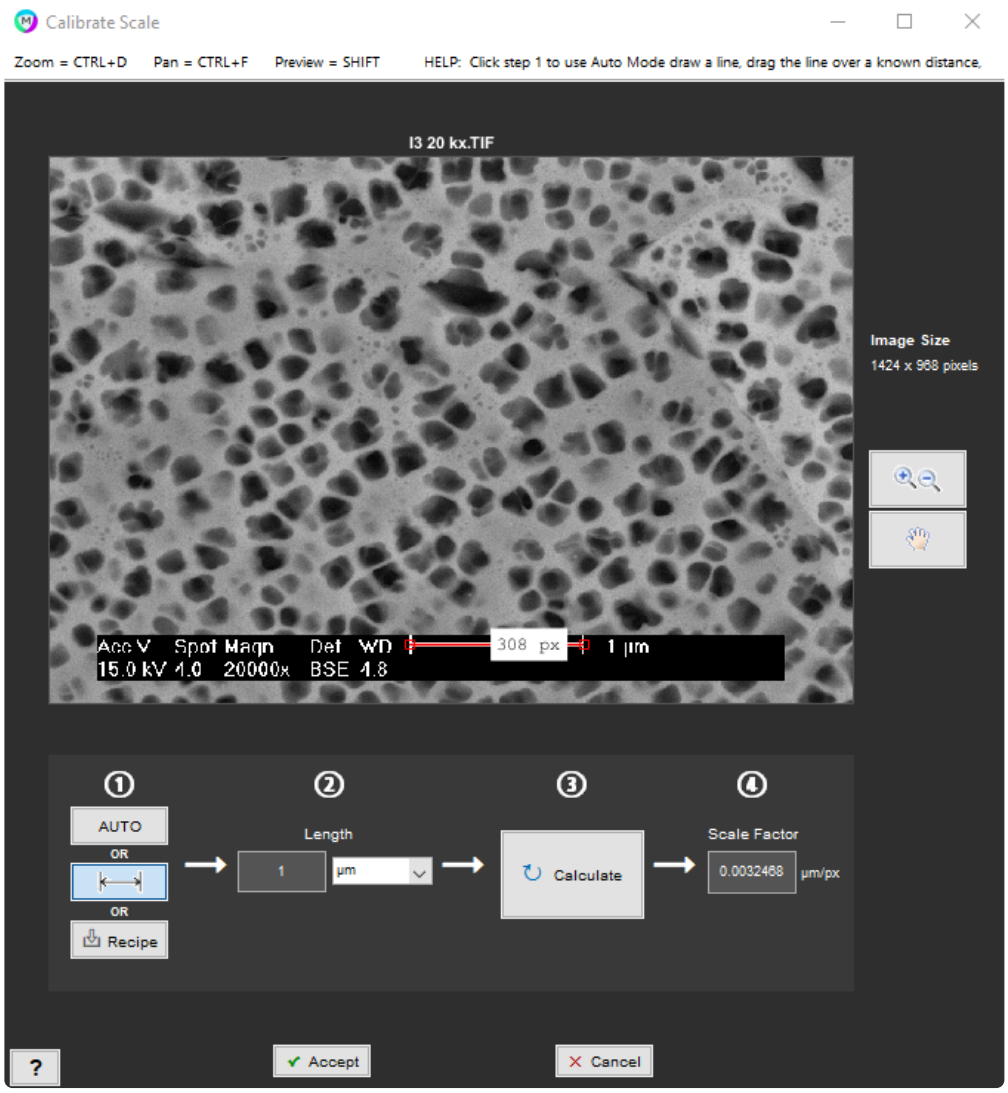
Auto Mode lets MIPAR automatically detect the scale bar in your image, without you having to manually define it! To use Auto Mode:

- Click “AUTO”
- Drag the blue box roughly around the area of the scale bar
- Click “✓” to confirm

MIPAR will then automatically detect the scale bar and place a line across it. You may then proceed with

step 2 and 3 (see below) to enter the physical distance and calculate the scale factor.



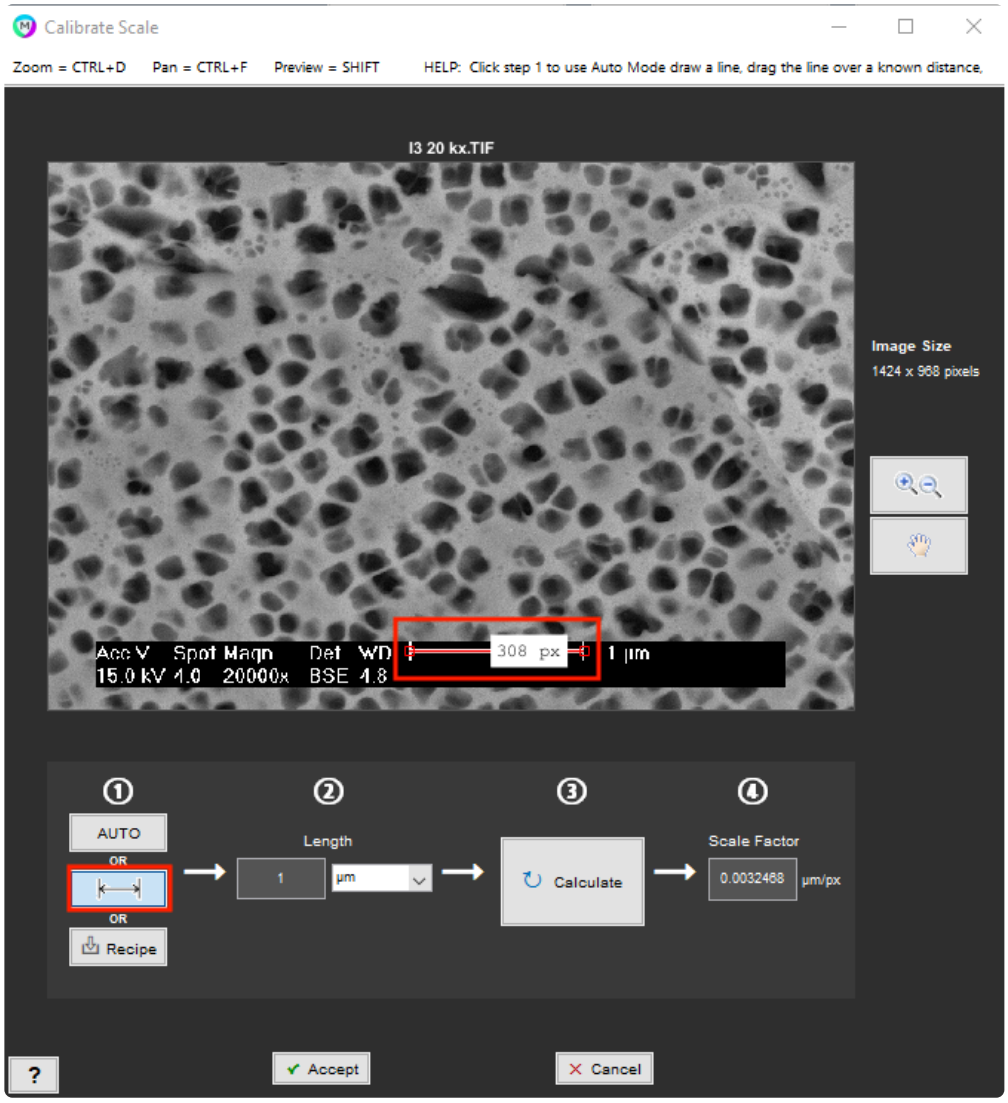


Manual Mode

Manual Mode may be used when MIPAR is unable to automatically detect the scale bar in your image. To use Manual Mode:

- Click the Distance Line tool
- Drag the line to span across the scale bar

You may then proceed with step 2 and 3 (see below) to enter the physical distance and calculate the scale factor.



Dynamic Mode

Dynamic Mode is used to determine the scale factor based on the size of a recognized feature. A recipe which uses Dynamic Mode is able to automatically adjust its scale factor per image based on the pixel size of the detected feature. Dynamic Mode accepts a “sub-recipe” for detection and a defined dimension to measure. When the main recipe is run, this sub-recipe first detects the scale-determining feature and sets the scale factor according to the feature’s measured pixel and defined physical dimensions.

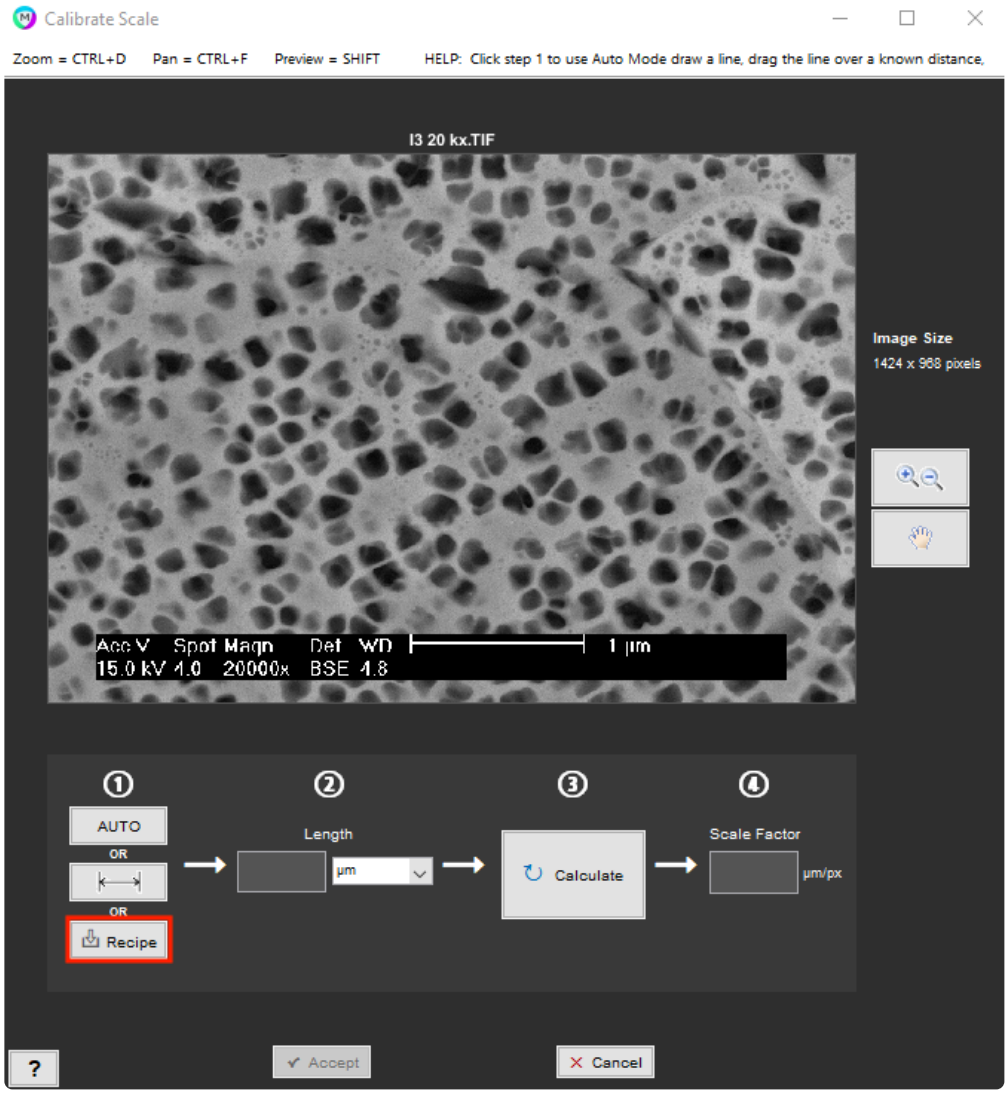
Recipe requirements:

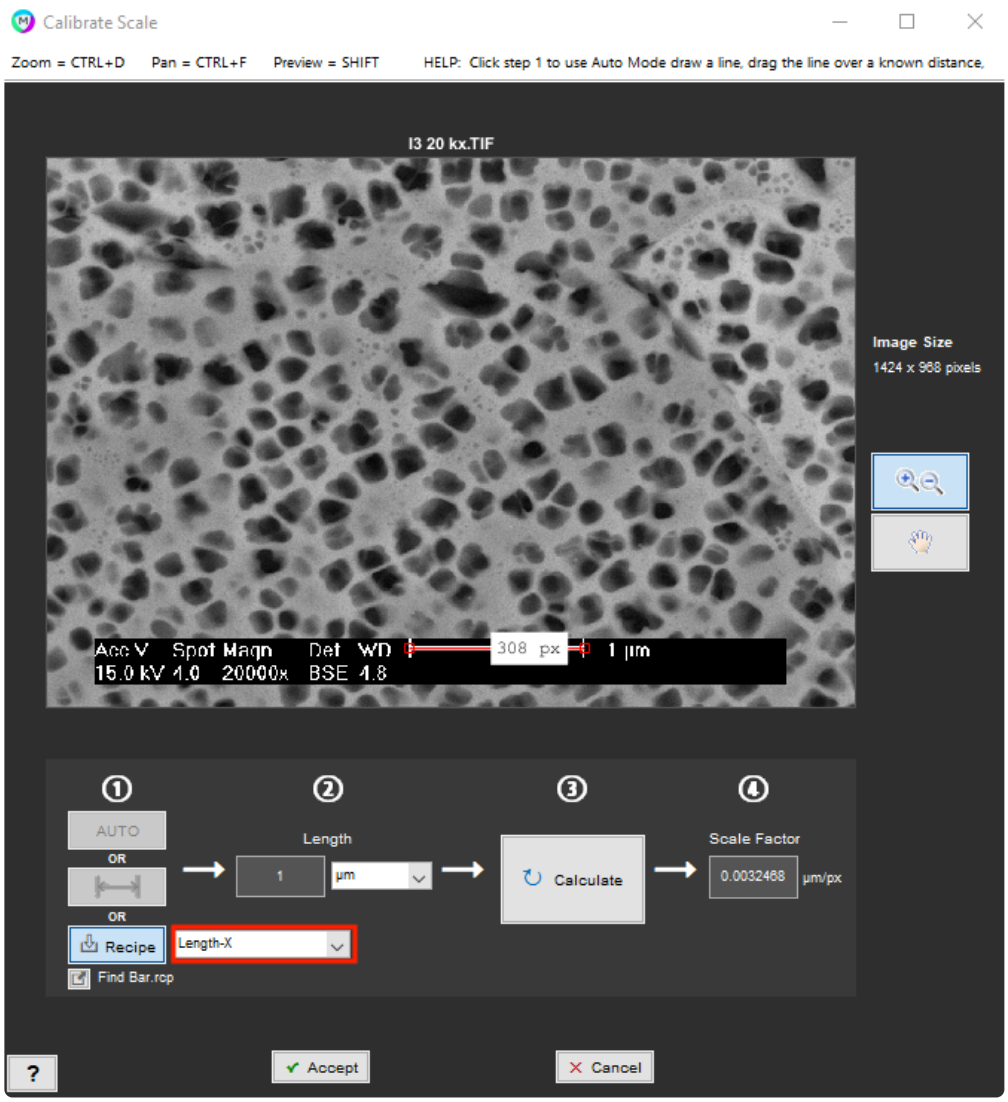
- Final step must be a selection (binary) image
- Recommended that final step only selects one feature (largest feature will be used if there are more than one)
- Any layers will be ignored (only final step considered)

To use Dynamic Mode:

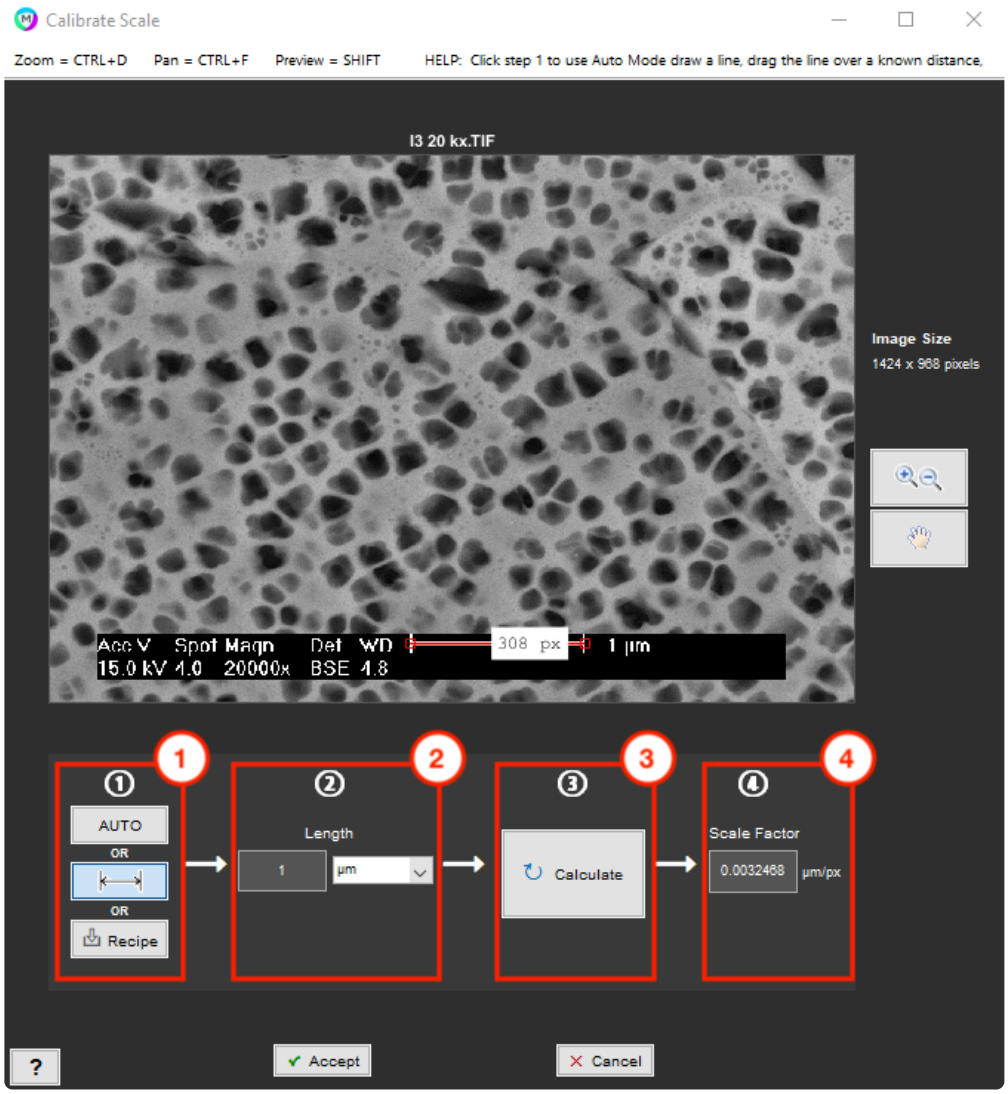
- Click the “Recipe” button > choose the recipe designed to detect feature to measure
- Select measurement of interest (line spanning feature will adjust automatically as measurement is changed)

You may then proceed with step 2 and 3 (see below) to enter the physical distance and calculate the scale factor.




Calibrate Scale
Zoom = CTRL+D
Pan = CTRL+F
Preview = SHIFT
HELP: Click step 1 to use Auto Mode draw a line, drag the line over a known distance.
I3 20 kx.TIF
Acc.V Spot Magn Det WD
15.0 kV 4.0 20000x BSE 4.8
308 px
1 µm
Image Size
1424 x 968 pixels
1
2
3
4
AUTO
OR
OR
Recipe
Length-X
Find Bar.rcp
Length
1
µm
Calculate
Scale Factor
0.0032468
µm/px
?
Accept
Cancel

General Use



1. Set Line

Use Auto Mode, Manual Mode (activate the Distance Line tool), or Dynamic Mode (“Recipe” button)

2. Enter Distance

Enter known distance spanned by line

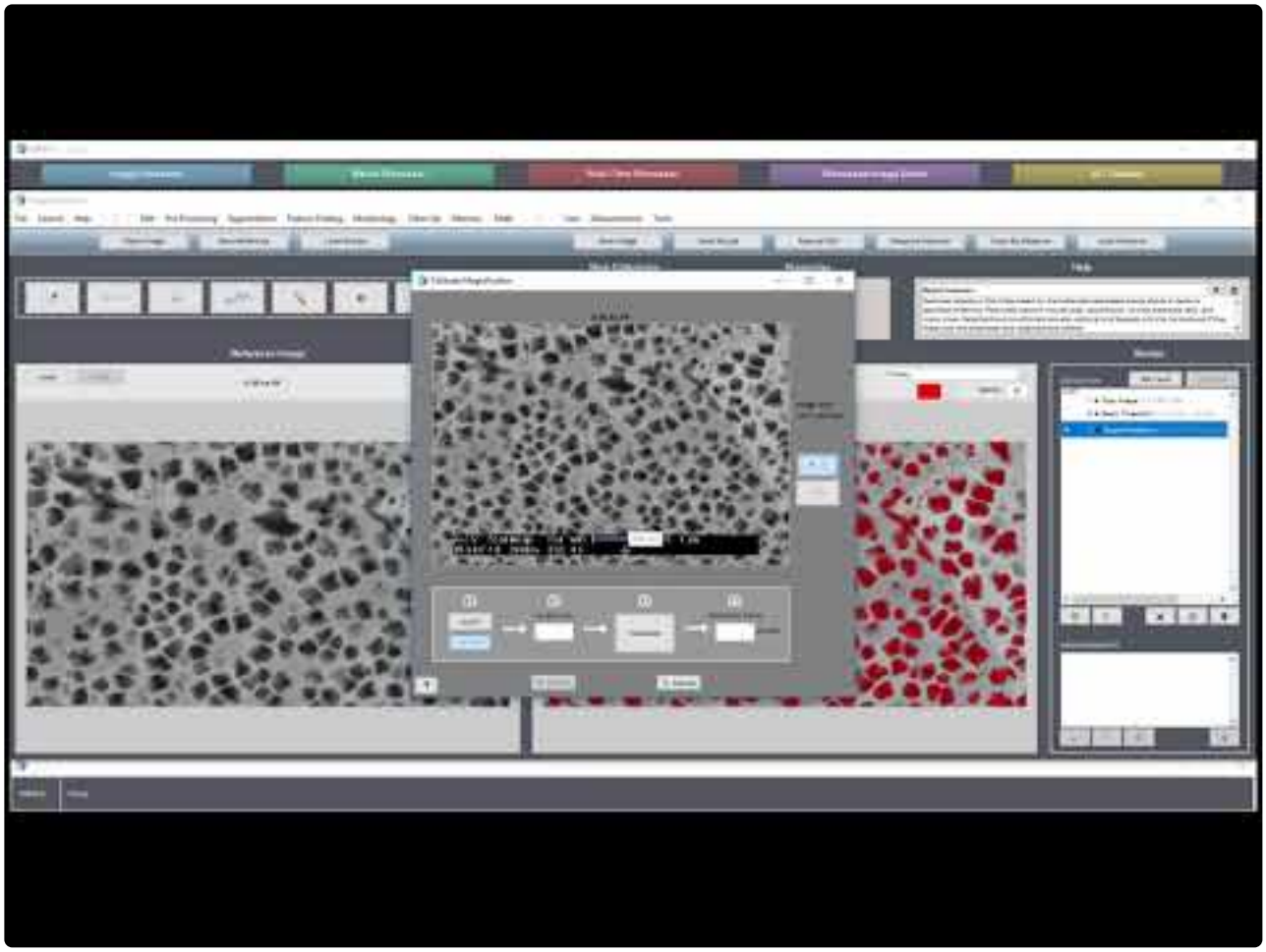
3. Calculate

Calculate the scale factor

4. Scale Factor

The scale factor is displayed. Click “Accept” or “Cancel”.

Tutorial



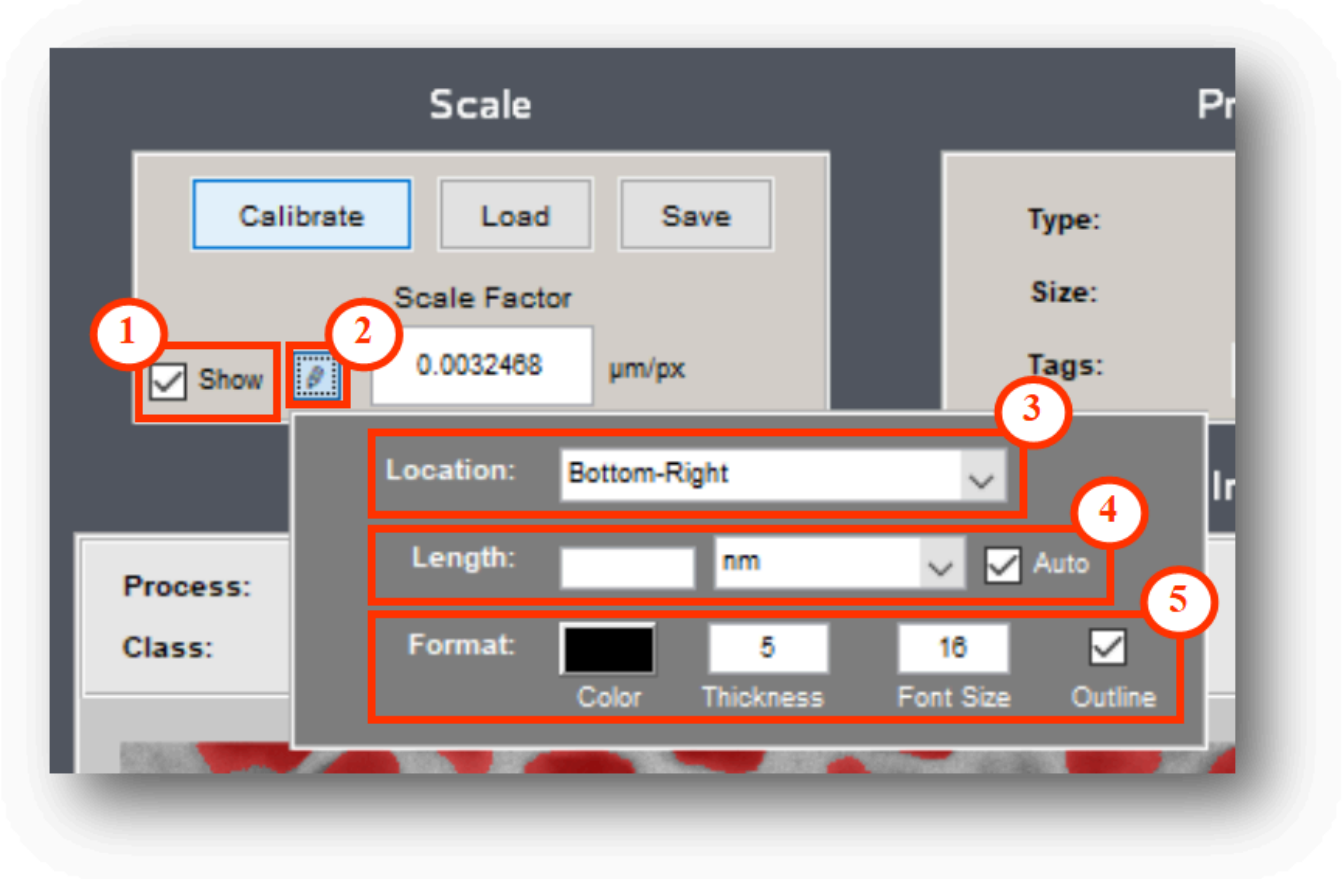
https://www.youtube.com/embed/il4U7UuiVbc?rel=0

https://www.youtube.com/embed/il4U7UuiVbc?rel=0

Scale Bar

Scale bar toggle allows users to overlay a scale bar based on parameters specified in the scale bar format settings.

Scale Bar Formatting



1. Show/Hide Scale Bar

2. Scale Bar Settings

3. Location

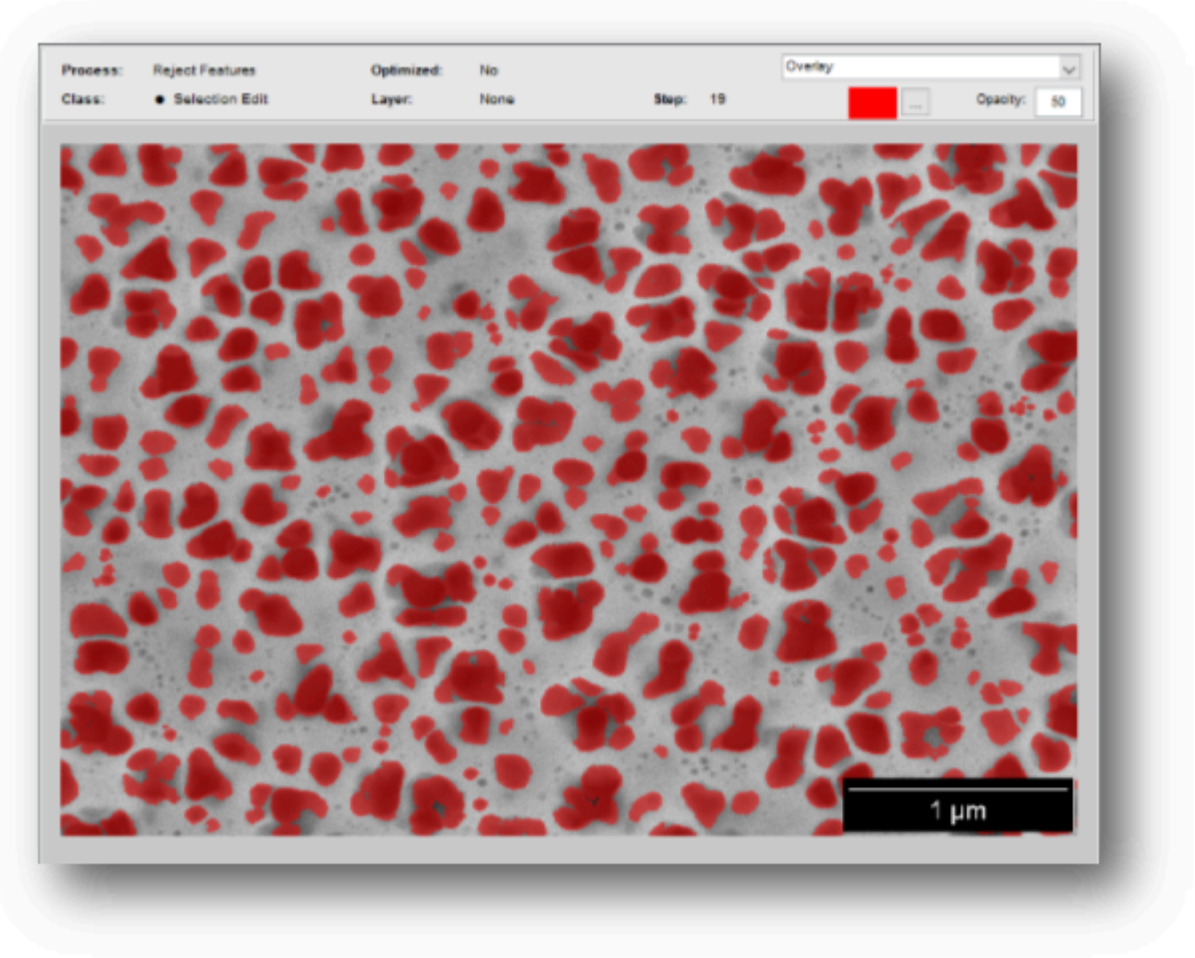
4. Length

Allows users to toggle between manually set scale bar length, an automatically generated scale bar as well are set scale bar units.

5. Formatting

- **Color**: Toggles between a black line on a white background and a white line on a black background color scheme.
- **Thickness**: User set scale bar thickness.
- **Font Size**: Font size of scale bar text.
- **Outline**: Outline toggle to accentuate the scale bar.

Example Scale Bar



Optimization

This interface allows you to setup and start recipe step (process) optimization (i.e., auto-setting determination). After selecting optimization parameters, you will be asked to enter a range for some of the process' parameters. This will start a loop which will execute the process at each parameter combination within the specified range. The parameter combination which yields the best quality metric will be set as the new parameter combination for the process. This calculation will be re-performed for any image the Recipe is run on.

The optimization interface is available by clicking the the "Optimize" button in the top right corner of the Recipe panel. The following steps are optimizable:

Pre-Processing

- Rotate Image
- Adjust Contrast
- Smart Cluster
- Interpolation

Segmentation

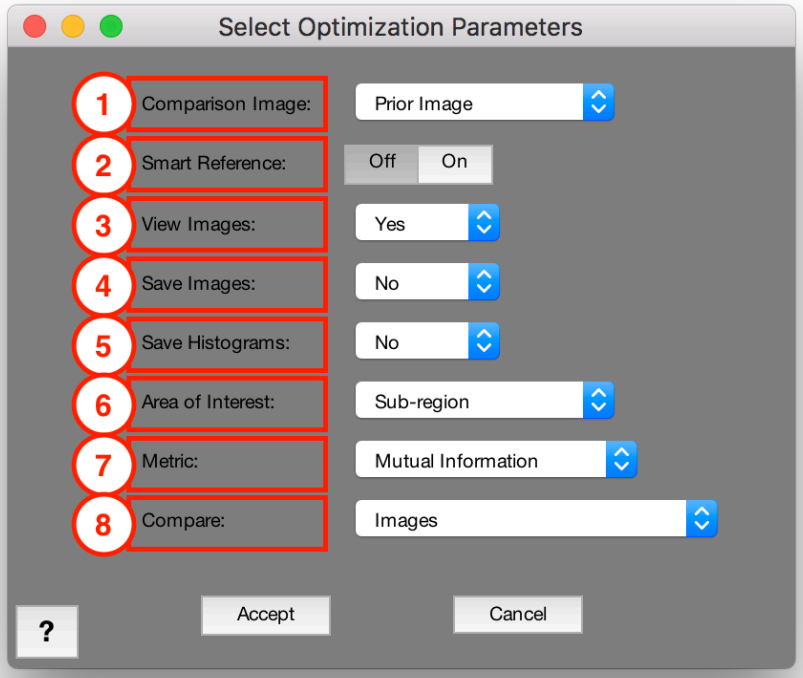
- Basic Threshold
- Range Threshold
- Adaptive Threshold
- E-M Threshold
- Watershed
- Local Threshold
- Region Grow
- Fast Marching Method
- Find Edges

Morphology

- Dilate Uniform
- Dilate Smart
- Dilate Retain
- Erode Uniform
- Erode Smart
- Erode Retain
- Separate Features
- Smooth Features

Clean-Up

- Reject Features



1. Comparison Image

Image to compare processed image to during optimization

2. Smart Reference

Choose whether to apply high-pass FFT filtering to comparison image. When “On” Smart Reference with make feature edges more important, and often improves the results of your optimization.

3. View Images

Choose whether to display optimization at each step

4. Save Images

Choose whether to save each frame of the optimization movie

5. Save Histograms

Choose whether to view/save the histograms of the processed image at each step

6. Area of Interest

Choose whether to use a sub-region or the entire image for optimization

7. Metric

Choose metric to use for comparing processed image to the comparison image

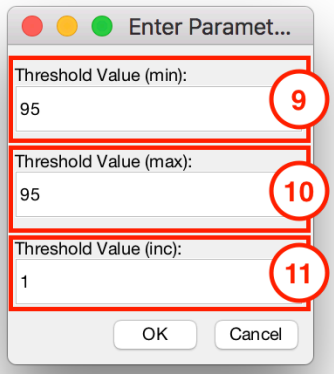
- **Selection Match:** A fast measure of image similarity that describes how well the processed image describes the comparison image. Only available when optimizing steps which result in a selection image. Does not take into account the uncertainty of the comparison image. Better suited for simpler segmentation problems with clear contrast between features and background. The optimization will choose the parameter combination which yields a maximum Selection Match.

- **Mutual Information:** A more advanced measure of image similarity that describes how well the processed image describes the comparison image [1,2]. Available when optimizing steps which result in either a selection or grayscale image. Takes into account the uncertainty of the comparison image. The optimization will choose the parameter combination which yields a maximum mutual information.

- **Intensity Match:** A measure of image similarity between two grayscale images. Only available when optimizing steps which result in a grayscale image. Computed as the normalized correlation coefficient between both images.

8. Compare

Choose whether to use the images, or their two-point correlation functions for comparison



9. Parameter Value (min)

Minimum value of the parameter range to explore

10. Parameter Value (max)

Maximum value of the parameter range to explore

11. Parameter Value (inc)

Increment (step size) to take between the minimum and maximum parameter values

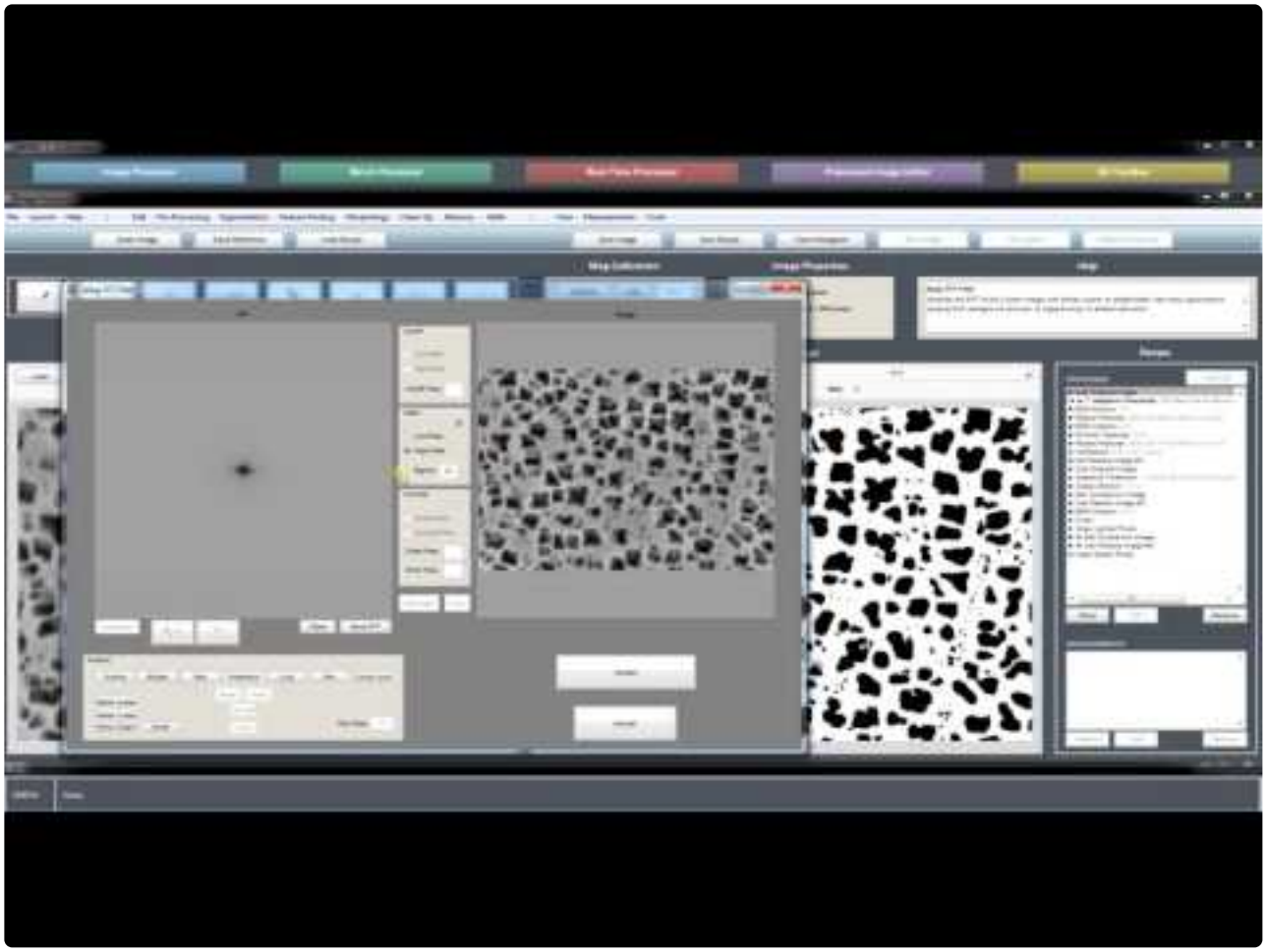
> ✻ Some functions allow for 2 or 3 parameters to be optimized simultaneously. In this case, additional sets of [min, max, inc] entries will be shown. Set the min and max equal to each other to fix any parameter during optimization.

References

[1] Rahunathan, Smriti, D. Stredney, P. Schmalbrock, and B.D. Clymer. Image Registration Using Rigid Registration and Maximization of Mutual Information. Poster presented at: MMVR13. The 13th Annual Medicine Meets Virtual Reality Conference; 2005 January 26–29; Long Beach, CA.

[2] D. Mattes, D.R. Haynor, H. Vesselle, T. Lewellen, and W. Eubank. “Non-rigid multimodality image registration.” (Proceedings paper).Medical Imaging 2001: Image Processing. SPIE Publications, 3 July 2001. pp. 1609–1620.

Tutorial



https://www.youtube.com/embed/HCIFGOyt6jI?rel=0

https://www.youtube.com/embed/HCIFGOyt6jI?rel=0

Measurements

Measure Image

Global measurements report one measurement per image. Common measurements include area fraction of features, total perimeter of features, and number of features. Click any link below for descriptions of the many global measurements available in MIPAR.

- Area
- Area Fraction
- Count
- Estimate Count
- Intercepts
- Image Dimensions
- Number Density
- Perimeter
- Perimeter Fraction
- *Intensity Mean
- *Intensity StdDev
- *Intensity Sum
- *Correlation Coefficient
- *Mutual Information

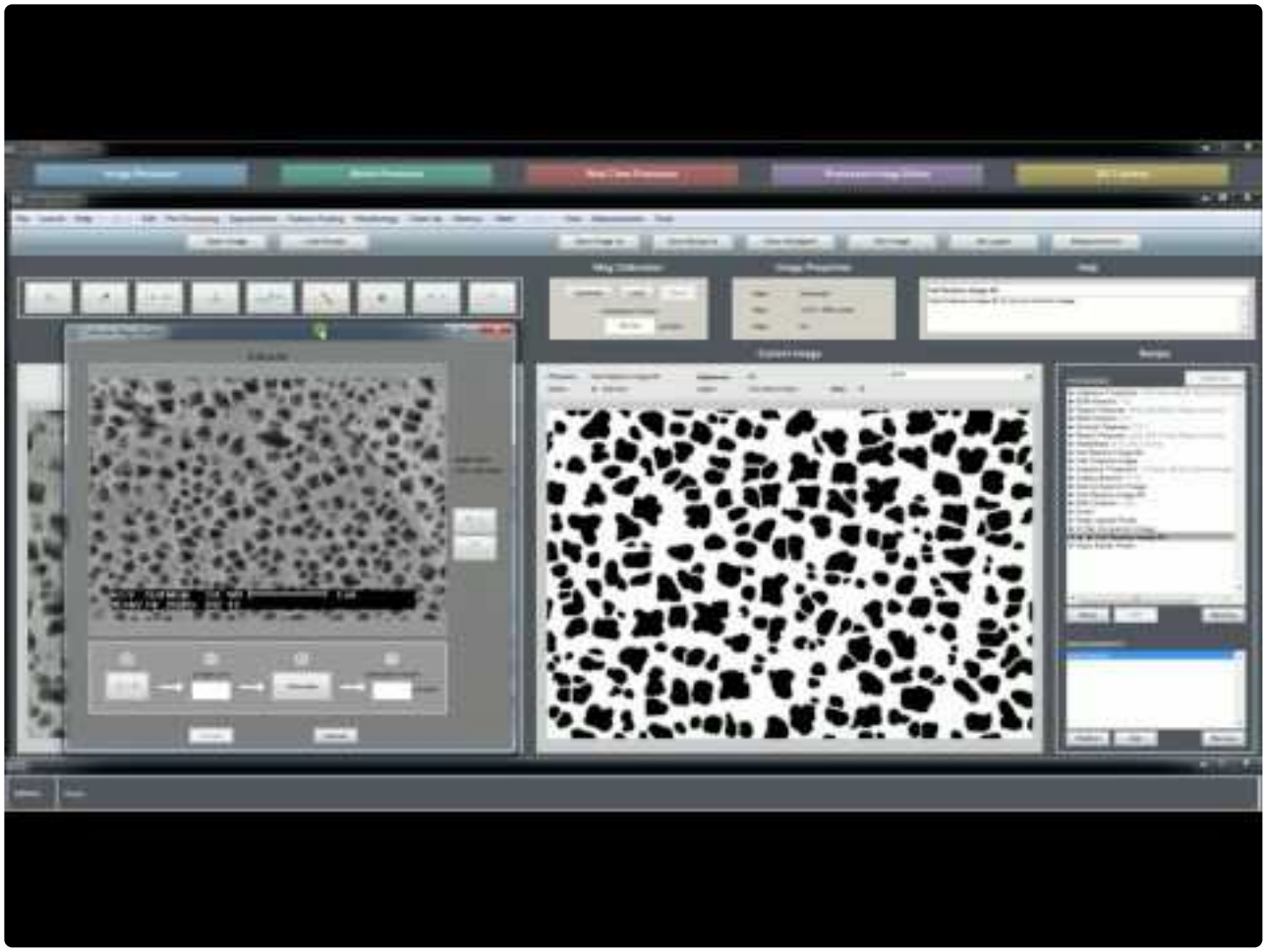
Measure Features

Feature measurements report one measurement for each feature in an image. Descriptions of the many feature measurements available in MIPAR can be found here.

Measure Local

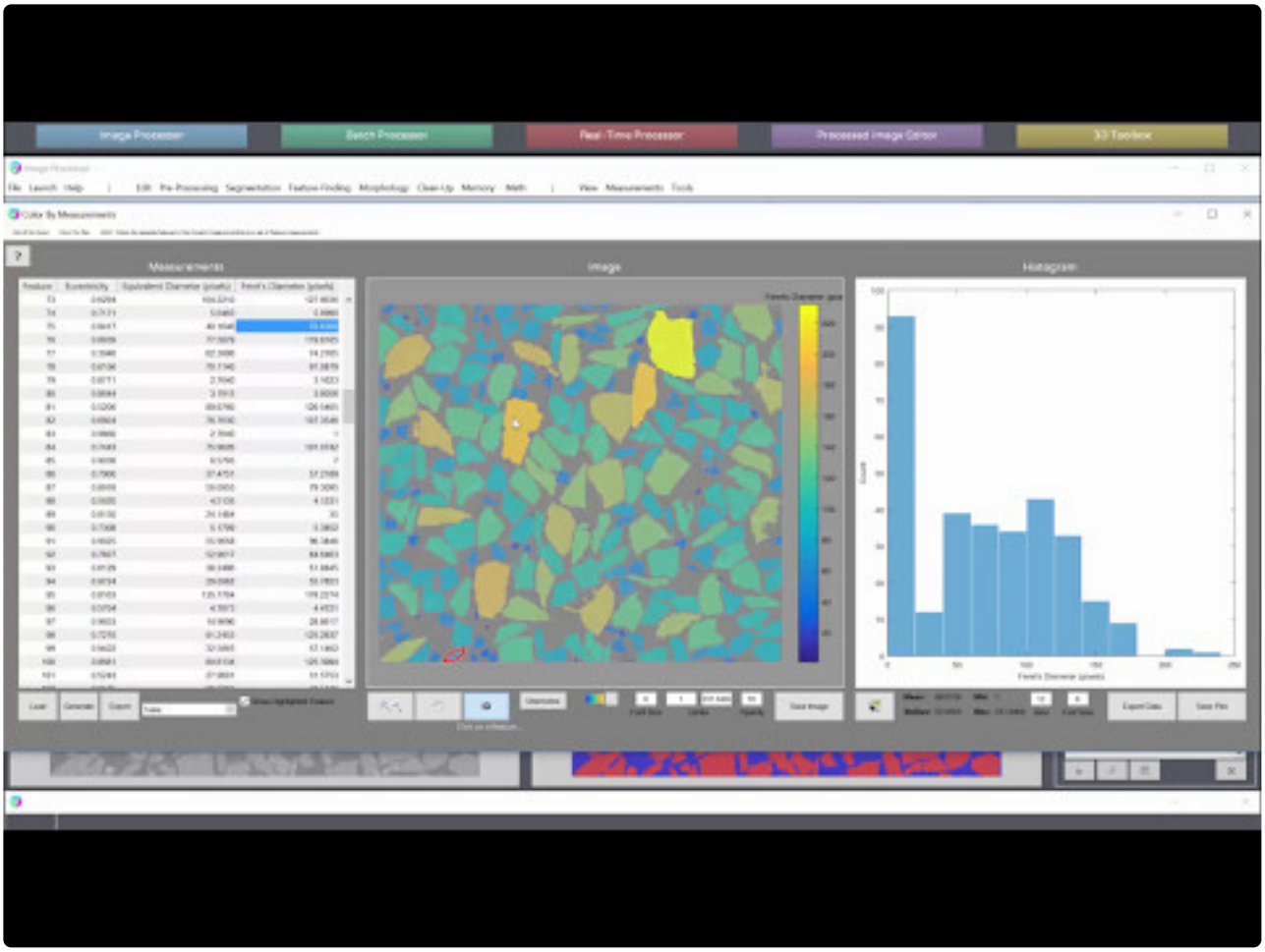
Local measurements report one measurement for each pixel. For some measurements, each pixel in the image is measured. For others, only pixels within features are measured. Common metrics include local area fraction, thickness, and mean grayscale value. A description of the Local Measurements tool and its available measurements can be found here.

Tutorials



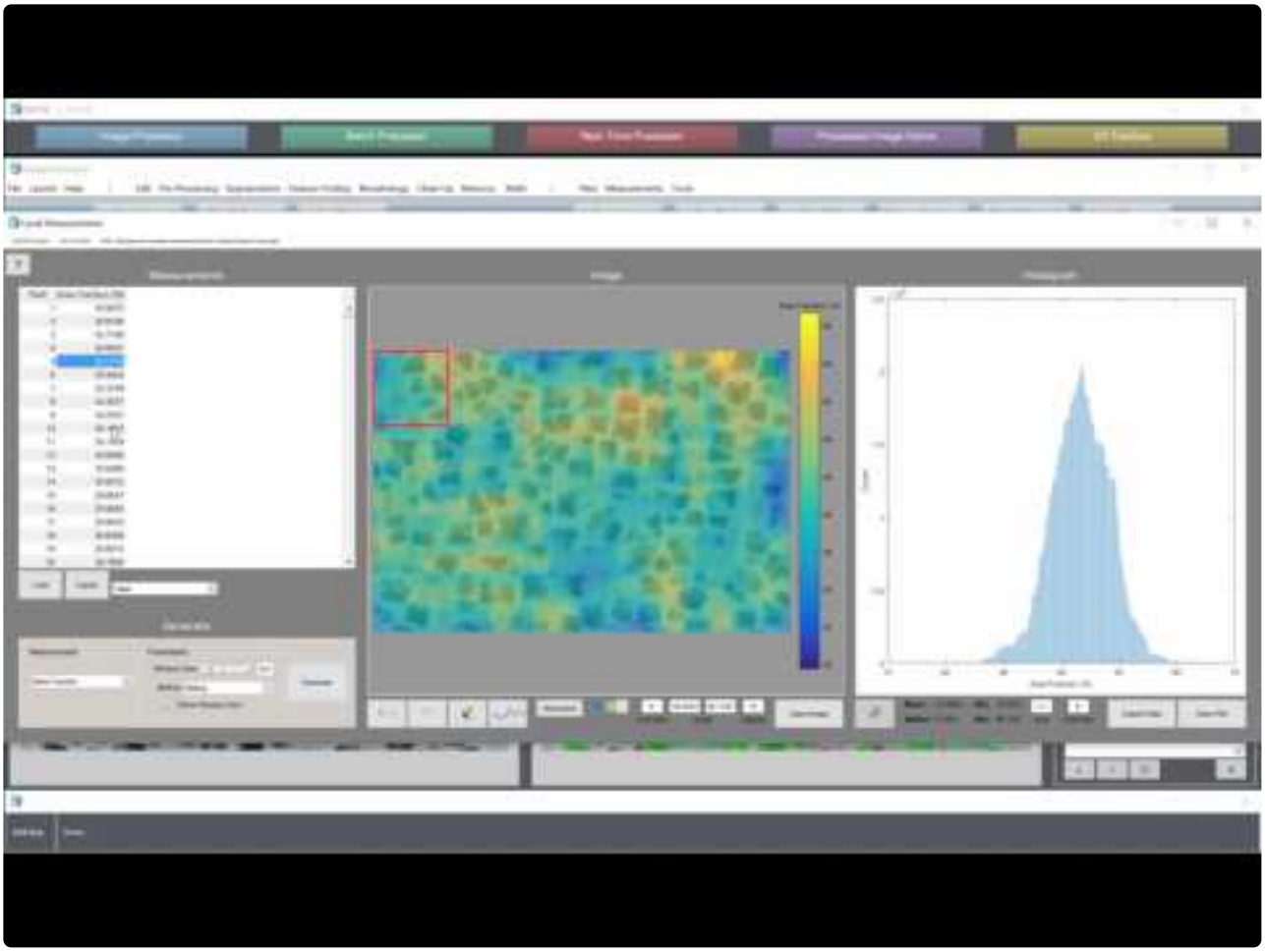
https://www.youtube.com/embed/m_Nd_ETglH4?rel=0

https://www.youtube.com/embed/m_Nd_ETglH4?rel=0



https://www.youtube.com/embed/kvhZrzKOubE?rel=0

https://www.youtube.com/embed/kvhZrzKOubE?rel=0



https://www.youtube.com/embed/WhAKluBliXs?rel=0

https://www.youtube.com/embed/WhAKluBliXs?rel=0

Functions

This section provides details on all functions available in MIPAR's Image Processor.

Each function's page contains the following information:

- **Title**
- **Path**
- **Summary**
- **Screenshot (if applicable)**
- **Parameters (if applicable)**
- **References (if applicable)**

Categories

- File
- Edit
- Color
- Pre-Processing
- Segmentation
- Morphology
- Clean-Up
- Memory
- Math
- View
- Measurements
- Tools

Note that many of these menus will not be displayed while in Simple Recipe Mode.

File

Contains functions for image opening and saving, as well as Recipe loading and saving.

Functions

- Open Image
- Open Recent Image
- Open Images as Channels
- Load Reference Image
- Load Recipe
- Load Recent Recipe
- Save Current Image
- Save Reference Image
- Save All Images
- Save All Layers
- Save Recipe
- Save Recipe As
- Print Current Image
- Preferences
- Quit

Open Image

File > Open

Opens an image. All common image formats and Bio-Formats are supported.

Supported Formats

- TIF
- JPG
- BMP
- PNG
- GIF
- Bio-Formats (over 140 formats)
  Requires downloading the Bio-Formats package and placing inside the "plugins" folder in your installation directory

Tips

You can drag and drop an image into the recipe or image panels at any time to open the image.

Open Recent Image

File > Open Recent Image

Holds the last five images opened, for easy access.

Clear Recent clears this list.

Open Images as Channels

File > Open Images as Channels

Allows for opening of multiple images into color channels. The individual color information can be accessed using Color > *Channel Operation*.

Only grayscale images are able to be combined to channels

Load Reference Image

File > Load Reference Image

Loads an image as the reference image. All common image formats are supported. Grayscale images will be read-in as 8-bit. Color images will offer a prompt which allows the user to decide how to convert the color image to 8-bit grayscale.

Load Recipe

File > Load Recipe

Loads and applies a Recipe to the Opened Image.

Tips

You can drag and drop a recipe into the recipe panel at any time to load the recipe.

Load Recent Recipe

Holds the last five recipes opened, for easy access.

Clear Recent clears this list.

Save Current Image

File > Save Current Image

Saves the Current Image as an 8-bit TIF-file. The image will be saved as it is shown in the Current Image viewer.

Save Reference Image

File > Save Reference Image

Saves the References Image as an 8-bit (if grayscale) or true-color (if color) TIF-file. The image will be saved as it is shown in the Reference Image viewer.

Save All Images

File > Save All Images

Saves an image from each Recipe step based on the current viewing mode.

Save All Layers

File > Save All Layers

Saves a B/W image for each Layer in the Recipe with the name [Original Image Name]_[Layer Name].

Save Recipe

File > Save Recipe

Save current Recipe as an RCP-file. The current processes, measurements, and calibration factor will be saved as into the Recipe.

Save Recipe As

File > Save As Recipe

Save current Recipe as an RCP-file with a specified name. The current processes, measurements, and calibration factor will be saved as into the Recipe.

Print Current Image

File > Print Current Image

Sends Current Image to the selected printer. The image will be printed as it is shown in the Current Image viewer.

Preferences

File > Load Recipe

Allows user to set MIPAR preferences.

Quit

Quits the Image Processor.

Edit

Contains functions for editing image dimensions and manipulating image position and rotation.

Functions

- Undo
- Burn Scale Bar
- Set as Reference Image

Size

- Crop Image
- Resize Image

Position

- Rotate Image
- Flip Image
- Translate Image
- *Register Image

Sample

- Sparsely Sample Image
- Draw Random Lines

Undo

Edit > Undo

Undoes the last Recipe edit. Recipe will re-execute from the last change-point after undoing.

Burn Scale Bar

Edit > Burn Scale Bar

Burns a scale bar to the Current Image. When the step is added, the shown scale bar will be burned. Scale bar settings for this step can be edited at any time. This current Scale Factor will be used when burning the scale bar.

More information on scale bar settings may be found on the Scale Bar page.

Set as Reference Image

Edit > Set as Reference Image

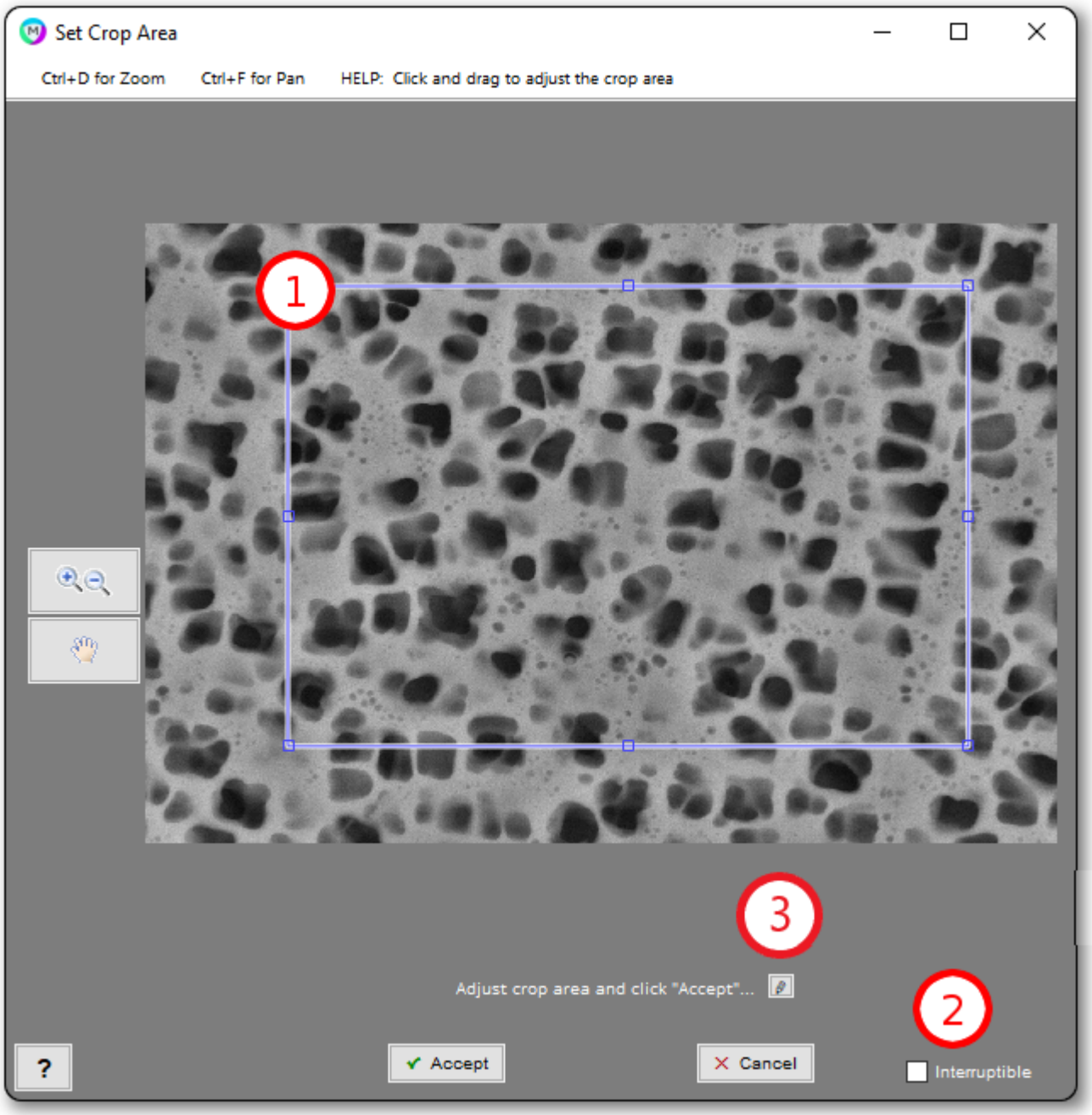
Set the Current Image as the Reference Image. Future B/W Current Images will then be overlaid, outlined, or labeled on this new Reference Image.

Crop Image

Edit > Crop Image

Crops the image to a sub-area of the previous image. Dimensions are listed as X: upper left corner, Y: upper left corner, with of crop box, height of crop box.

Draw



1. Adjust crop area

Click and drag the blue box to adjust the crop area. Click “Accept” when finished.

2. Interruptible in batch

If this box is checked and the recipe is run in batch, recipe execution will pause and the crop dialog will automatically open, allowing manual adjustment of the crop area for each image. Clicking “Accept” will cause recipe execution to resume.

3. Enter Dimensions

Manually enter or adjust the desired crop box dimensions as the (X,Y) position of the upper left corner, and the box width and height.

Auto

Crops the image to the perimeter of the features nearest to the image edges.

Apply Last

Requires a previous Crop Image step

Crops the image using the area defined in the last Crop Image step in the recipe.

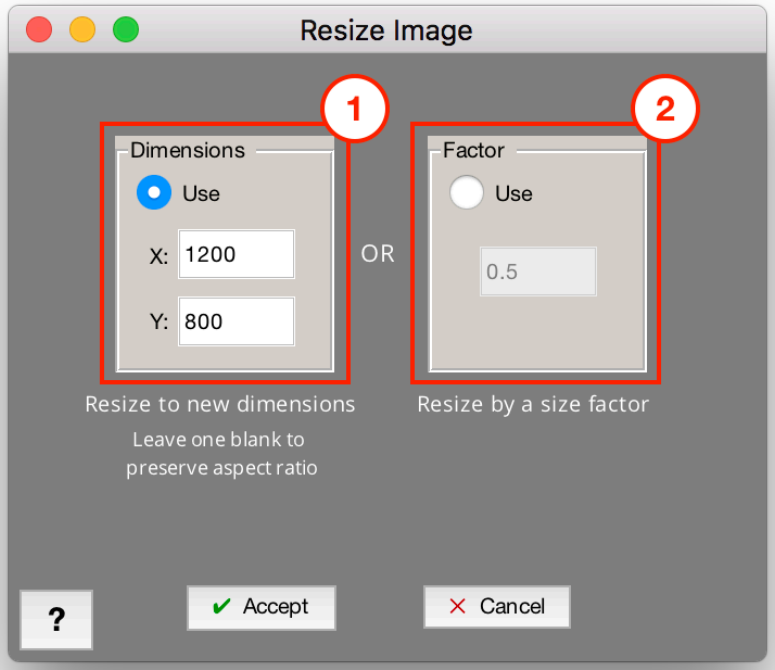
Tips

- Recipe building is smoothest when all image cropping and resizing is done in the beginning of the recipe.
- When working with large images, to improve performance, crop the image, optimize the recipe and remove the crop step when complete.

Resize Image

Edit > Resize Image

Resizes the image to the new dimensions, or by a size factor.



1. Dimensions

Resize image to new dimensions

- **X:** Enter new X-dimension. Leave blank (if Y is not) to maintain aspect ratio.
- **Y:** Enter new Y-dimension. Leave blank (if X is not) to maintain aspect ratio.

2. Size Factor

Resize by a multiplicative size factor (e.g., a size factor of 0.5 will reduce both X and Y dimensions by half)

Automatic Scale Adjustment

- In the Image Processor, when adding or removing a Resize step from a recipe, the scale will

automatically adjust to compensate.

- In the Batch Processor, when loading an image where its scale is automatically read from its metadata, this scale will automatically be adjusted by the presence of Resize steps in the recipe.
- These adjustments apply to the entire recipe, even steps before the Resize step.

Tips

• Recipe building is smoothest when all image cropping and resizing is done in the beginning of the recipe.

• Depending on the resolution of the features you're interested in segmenting, you can often drastically improve performance by down-sampling your image. This is recommended only if there is an insignificant loss of detail in the features of interest.

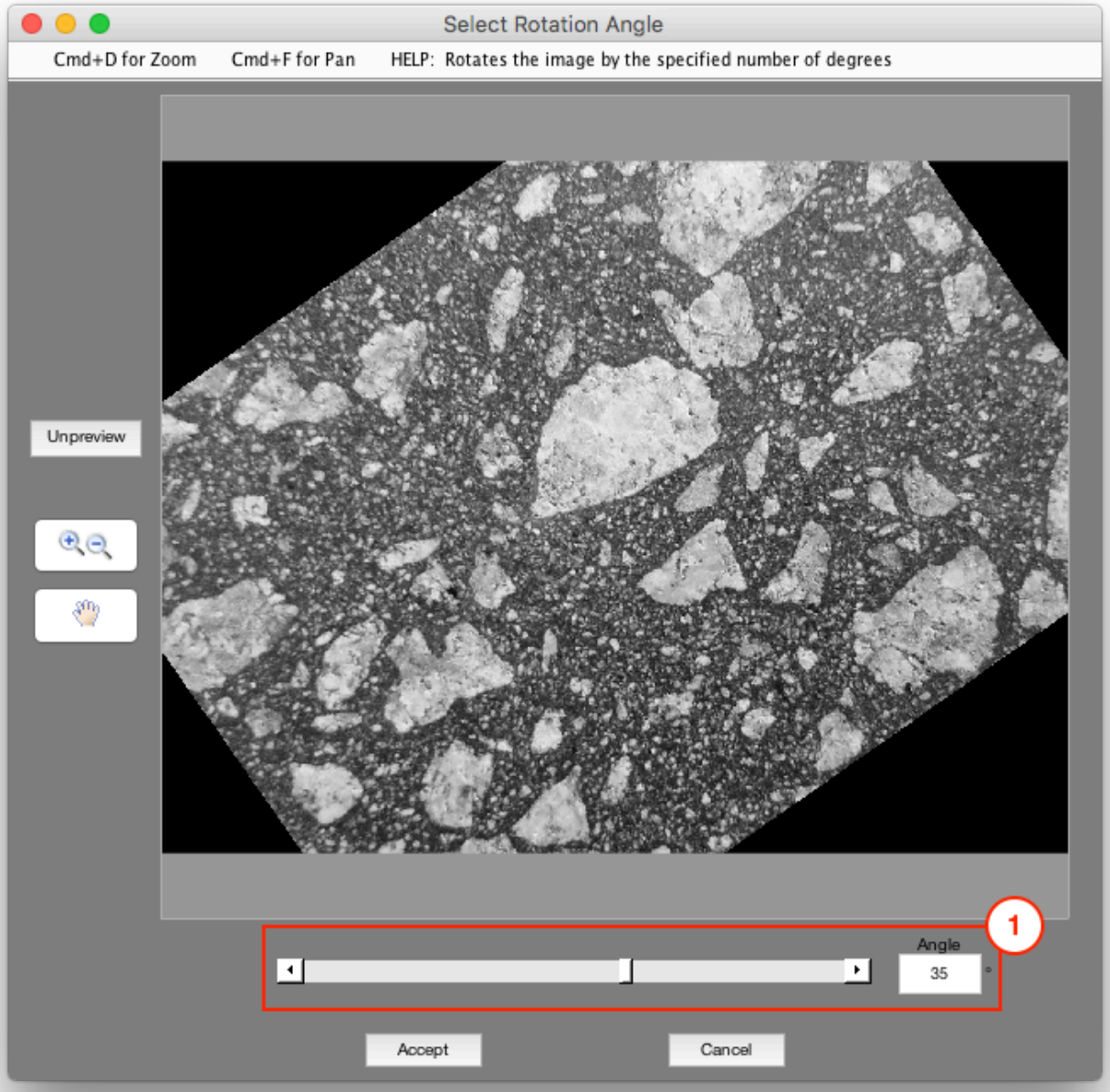
• Any steps involving image 'Math' will require both images to be the same size

Rotate Image

Edit > Rotate Image

Rotates the image by the specified number of degrees. Positive rotation is counterclockwise and negative is clockwise. Rotated image is clipped to maintain image frame dimensions.

Pixels that end up outside the image's new location are set to 0, resulting in black regions in a grayscale image and selected regions in a binary image.



1. Angle

Angle to rotate image by. Positive rotations are counterclockwise and negative clockwise.

Flip Image

Edit > Flip Image

Over Horizontal

Flips the image over the horizontal axis.

Over Vertical

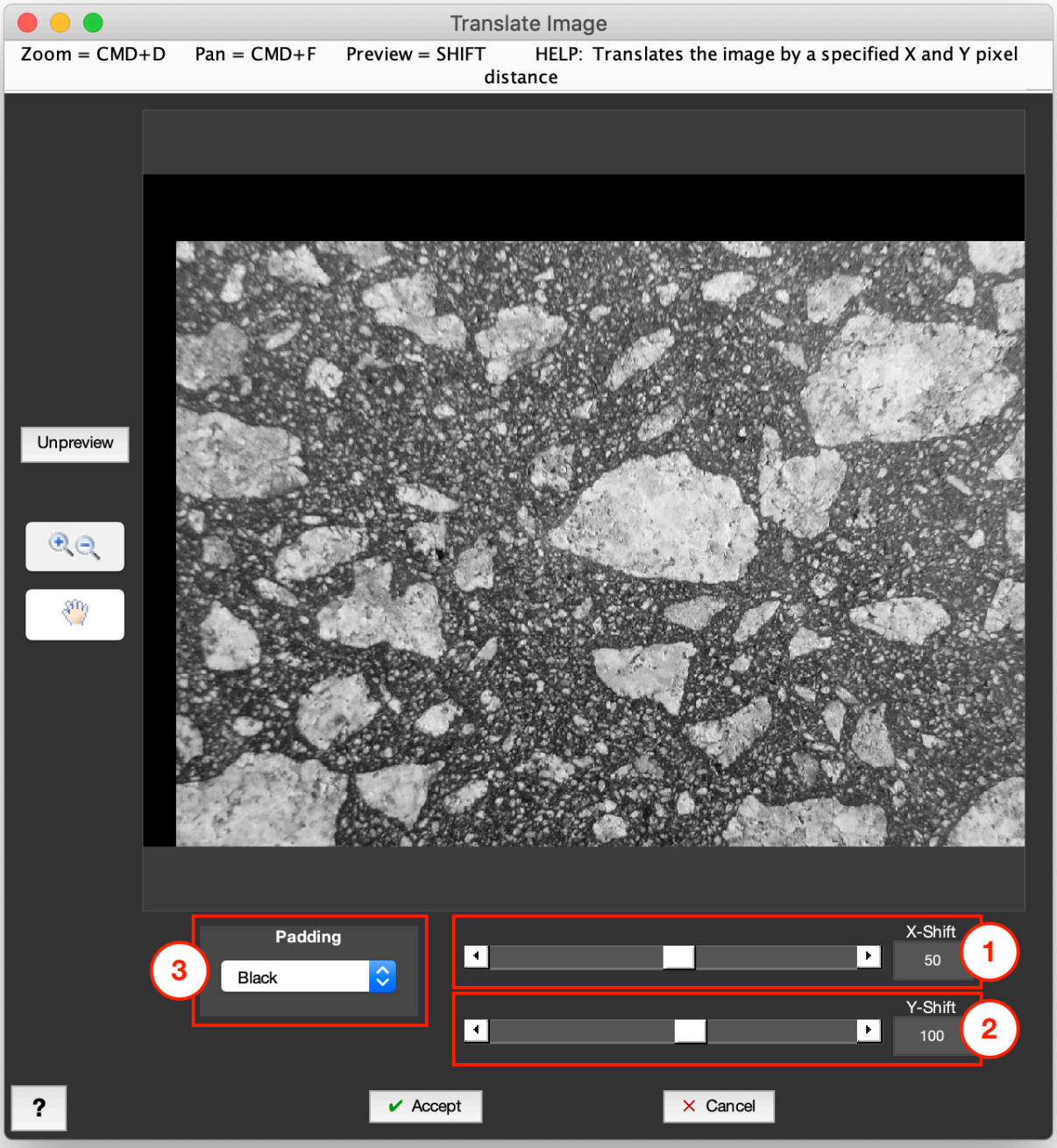
Flips the image over the vertical axis.

Translate Image

Edit > Translate Image

Translates the image by a specified X and Y pixel distance.

Pixels that end up outside the image’s new location are set to 0, resulting in black regions in a grayscale image and selected regions in a binary image.



1. X-Shift

Pixels to shift in X-direction

2. Y-Shift

Pixels to shift in Y-direction

3. Padding

Choose to use black or white padding when translating the image

Tips

- You can create image borders to use as a ‘Companion’ by selecting: Blank > Translate Image.
- Example: Blank > Translate: can generate a 1 pixel boundary line that you can use to isolate features touching a single edge.

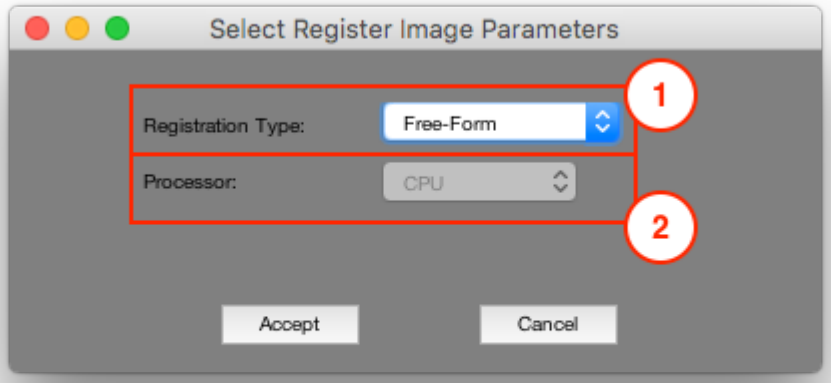
*Register Image

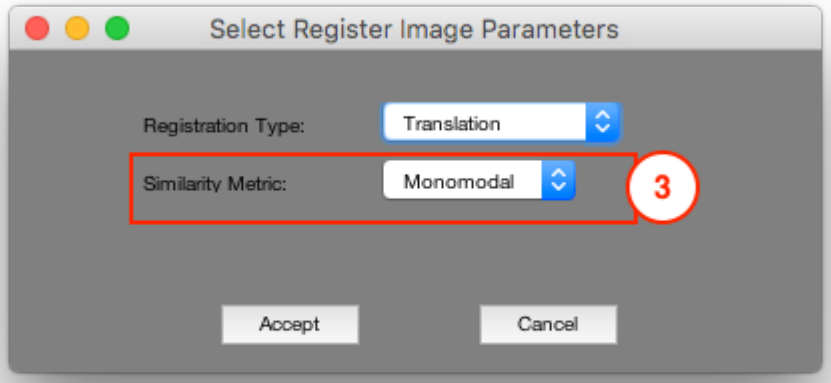
Edit > *Register Image

*Register Image

Requires Companion Image

Aligns the Current Image to the Companion Image using either a translation, rigid, similarity, affine, or free-form registration.





1. Registration Type

- **Free-Form:** Allows full non-rigid image distortions

- **Translation:** Allows only image translation in X and Y
- **Rigid:** Allows image translation and rotation
- **Similarity:** Allows image translation, rotation, and uniform scaling
- **Affine:** Allows image translation, rotation, uniform scaling, non-uniform scaling, and shearing

2. Processor *(for Registration Type "Free-Form")*

Not used currently

3. Similarity Metric *(for all other Registration Types)*

- **Monomodal:** Used for aligning two images of the same contrast type
- **Multimodal:** Used for aligning two images of different contrast type

Apply Last

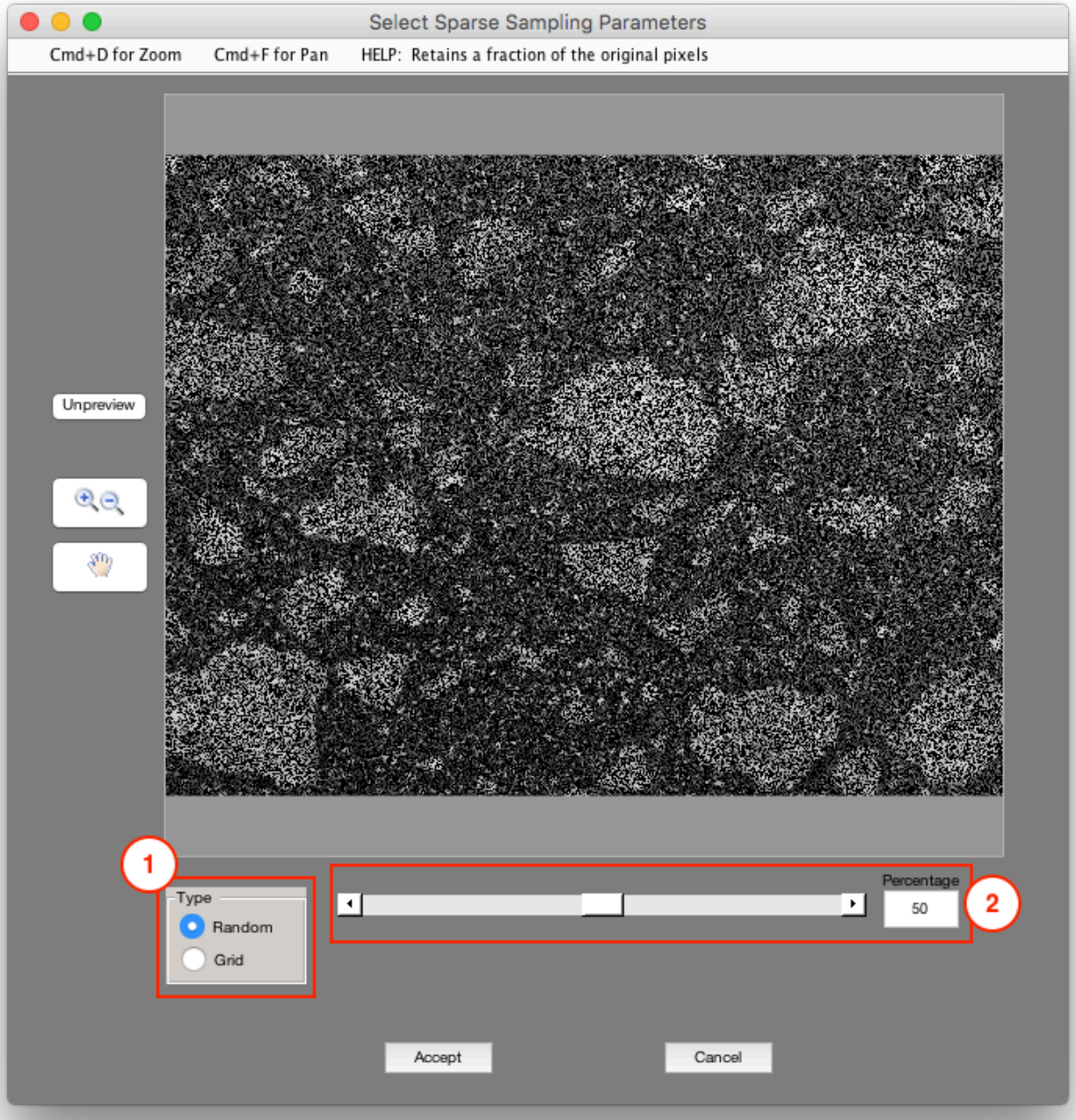
Requires a previous Register Image step

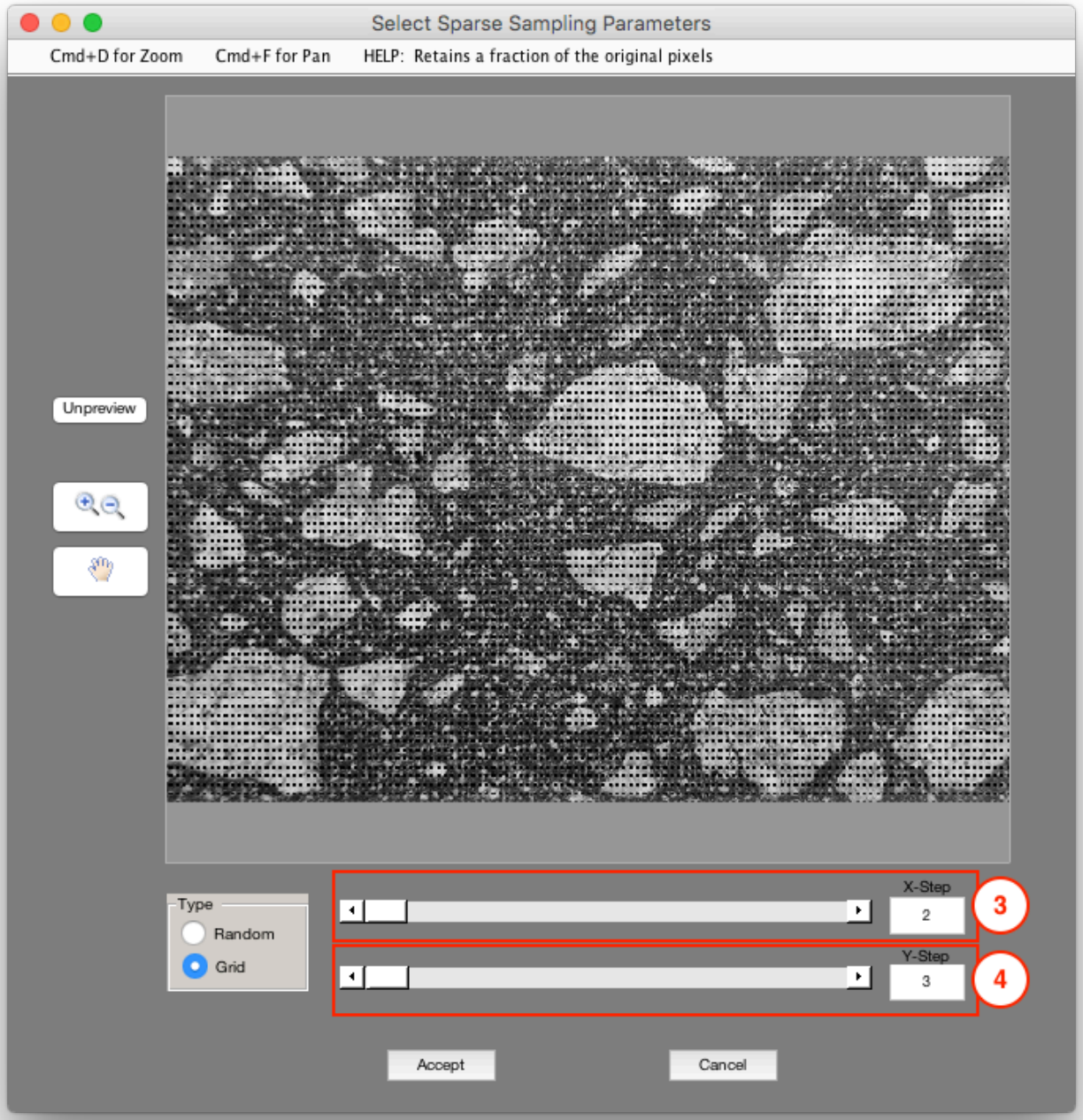
Aligns the Current Image to the Companion Image using the registration determined by the last Register Image step in the recipe.

Sparsely Sample Image

Edit > Sparsely Sample Image

Retains a fraction of the original pixels. Can be done randomly with a specified retained percentage or in a regular grid with specified pixel spacing.





1. Method

- **Random:** Use random sampling to produce sparse image
- **Grid:** Sparsely sample using regularly-spaced grid

2. Percentage

Percentage of pixels to keep

3. X-Step

X-step size of regularly-spaced grid

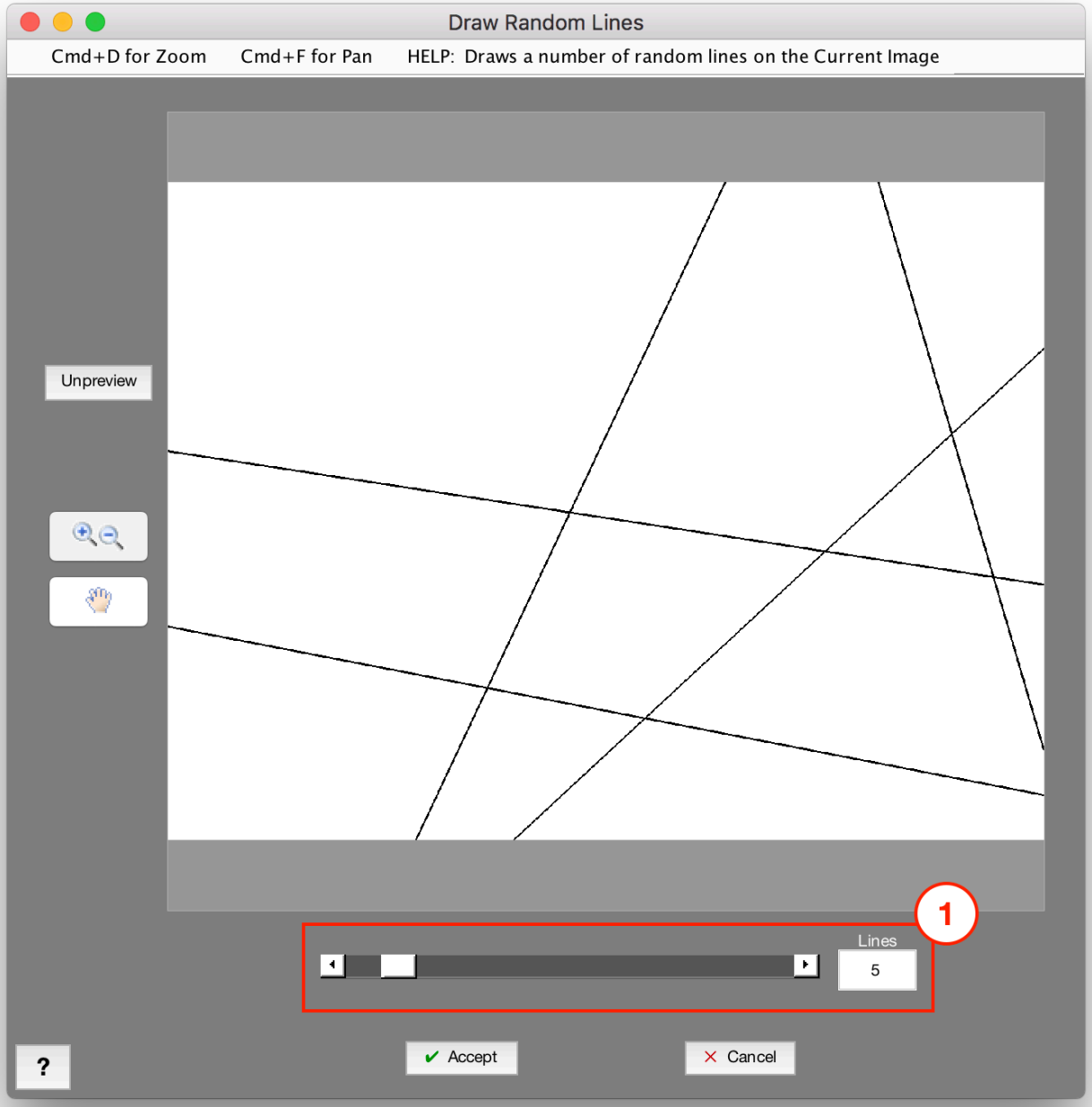
4. Y-Step

Y-step size of regularly-spaced grid

Draw Random Lines

Edit > Draw Random Lines

Draws a number of random lines on the Current Image.



1. Lines

Number of random lines to draw.

Color

Contains functions that operate on colored original images, producing grayscale Current Images from combinations of the various color channels.

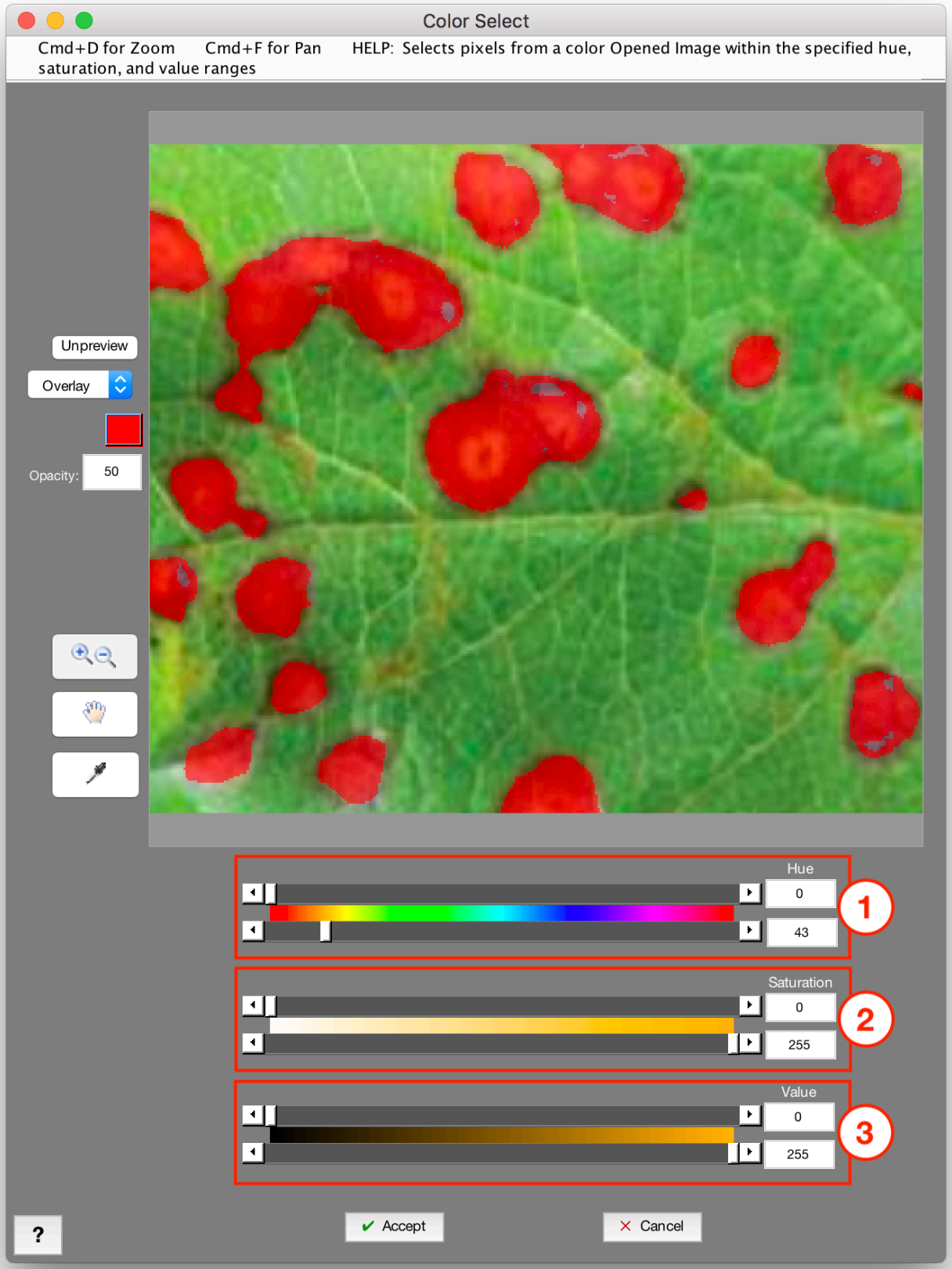
Functions

- Color Select
- Color Cluster
- Color Deconvolution
- Channel Operation

Color Select

Color > Color Select

Selects pixels from a color Opened Image within the specified hue, saturation, and value ranges.


Color Select
Cmd+D for Zoom Cmd+F for Pan HELP: Selects pixels from a color Opened Image within the specified hue, saturation, and value ranges
Unpreview
Overlay
Opacity:
50
Hue
0
43
1
Saturation
0
255
2
Value
0
255
3
?
Accept
Cancel

1. Hue

Sets the hue selection range. When the first limit is less than the second, the range between the two limits will be selected. When the first limit is greater than the second, the selection will be above the first limit, passed 360, to the second limit (selecting all the red hues).

2. Saturation

Sets the saturation selection range.

3. Value

Sets the value selection range.

Tips

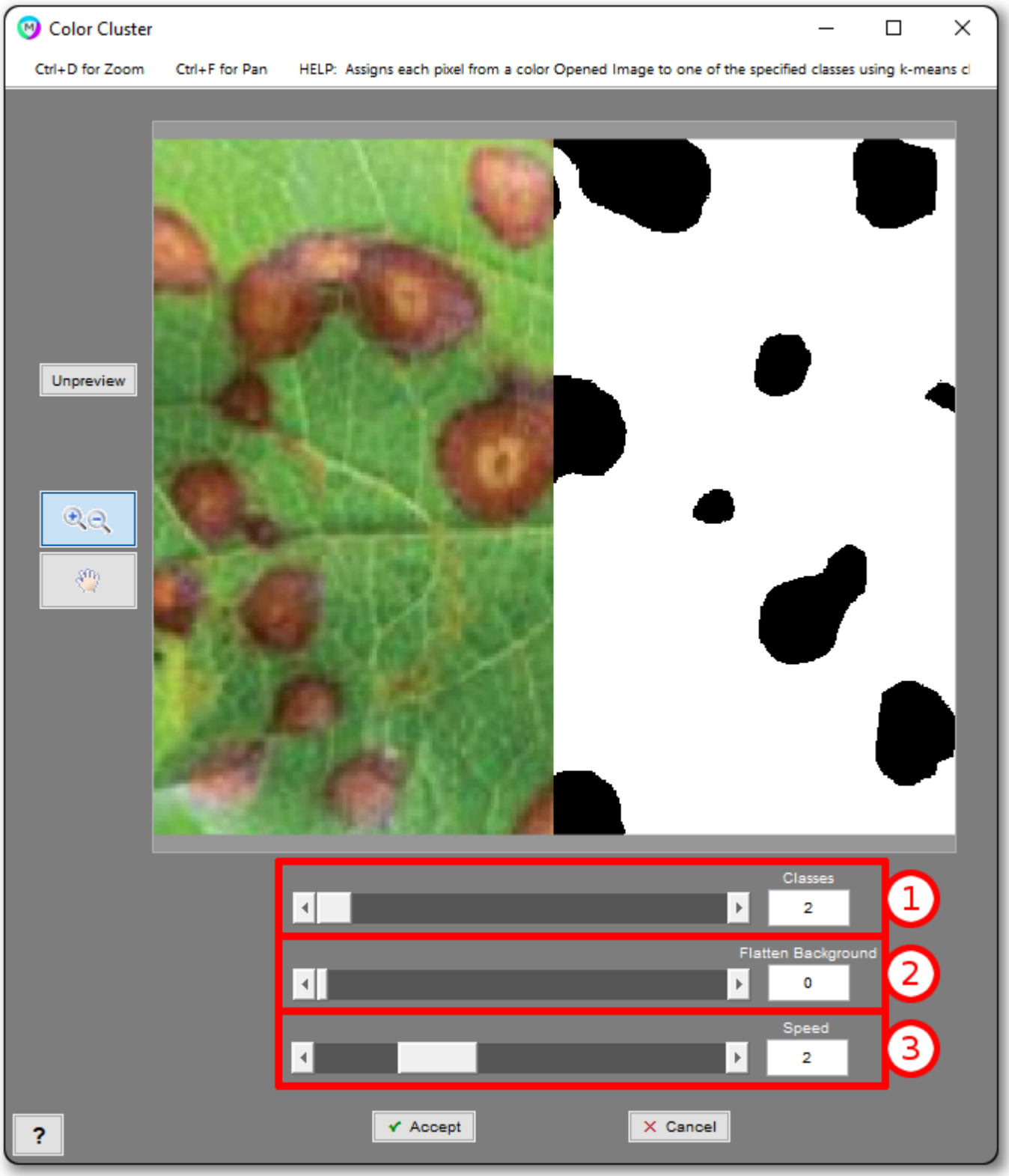
- Color Select is always applied to the original color image

Color Cluster

Color > Color Cluster

Assigns each pixel from the color Opened Image to one of the specified classes using k-means clustering [1-4]. This can be a powerful and objective way of segmenting color images. Works well for color images with 2 or more feature types of interest, where different colors define the different feature types. This function results in a grayscale image where each class is colored according to the class which it belongs.

To split the resulting image into different Layers, add a Set Memory image step afterwards, then iteratively call that restore image and use “Range Threshold” steps to select the different classes as B/W images.



1. Classes

Number of classes to cluster each pixel into. If you are trying to segment one set of features from the background, you should enter 2 classes.

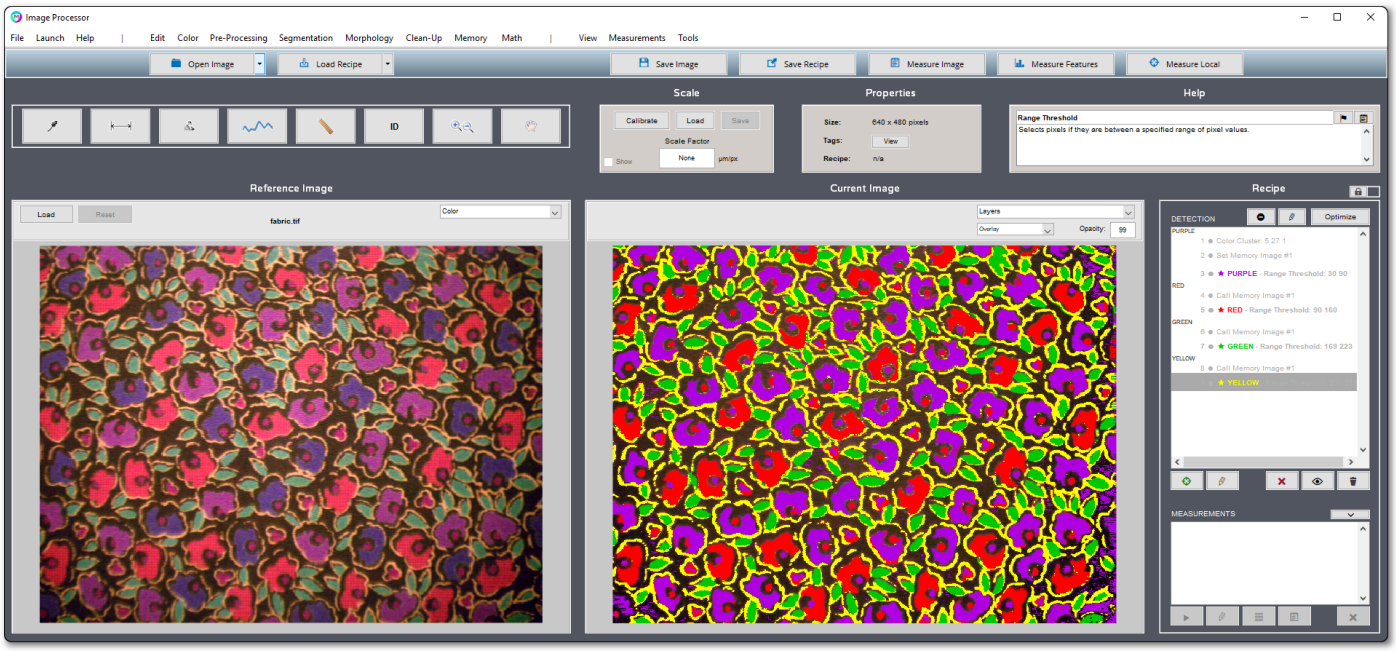
2. Flatten Background

Controls the window size used for background flattening prior to clustering. Use 0 for no background flattening.

3. Speed

Parameter that controls the resolution of the cluster. Lower speed results in higher resolution clustering, but takes longer. Higher speed results in faster execution but lower resolution clustering.

Example



Example Recipe following a *Color Cluster* step which selects each class and sets it as a Layer.

References

[1] Arthur, David, and Sergi Vassilvitskii. “K-means++: The Advantages of Careful Seeding.” SODA ‘07: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms. 2007, pp. 1027–1035.

[2] Lloyd, Stuart P. “Least Squares Quantization in PCM.” IEEE Transactions on Information Theory. Vol. 28, 1982, pp. 129–137.

[3] Seber, G. A. F. Multivariate Observations. Hoboken, NJ: John Wiley & Sons, Inc., 1984.

[4] Spath, H. Cluster Dissection and Analysis: Theory, FORTRAN Programs, Examples. Translated by J. Goldschmidt. New York: Halsted Press, 1985.

Tips

- Color Cluster is always applied to the original color image
- Works best when the features of interest are further away from each other on the color spectrum

Tutorials

https://www.youtube.com/embed/xdrw2Utgb2c?rel=0

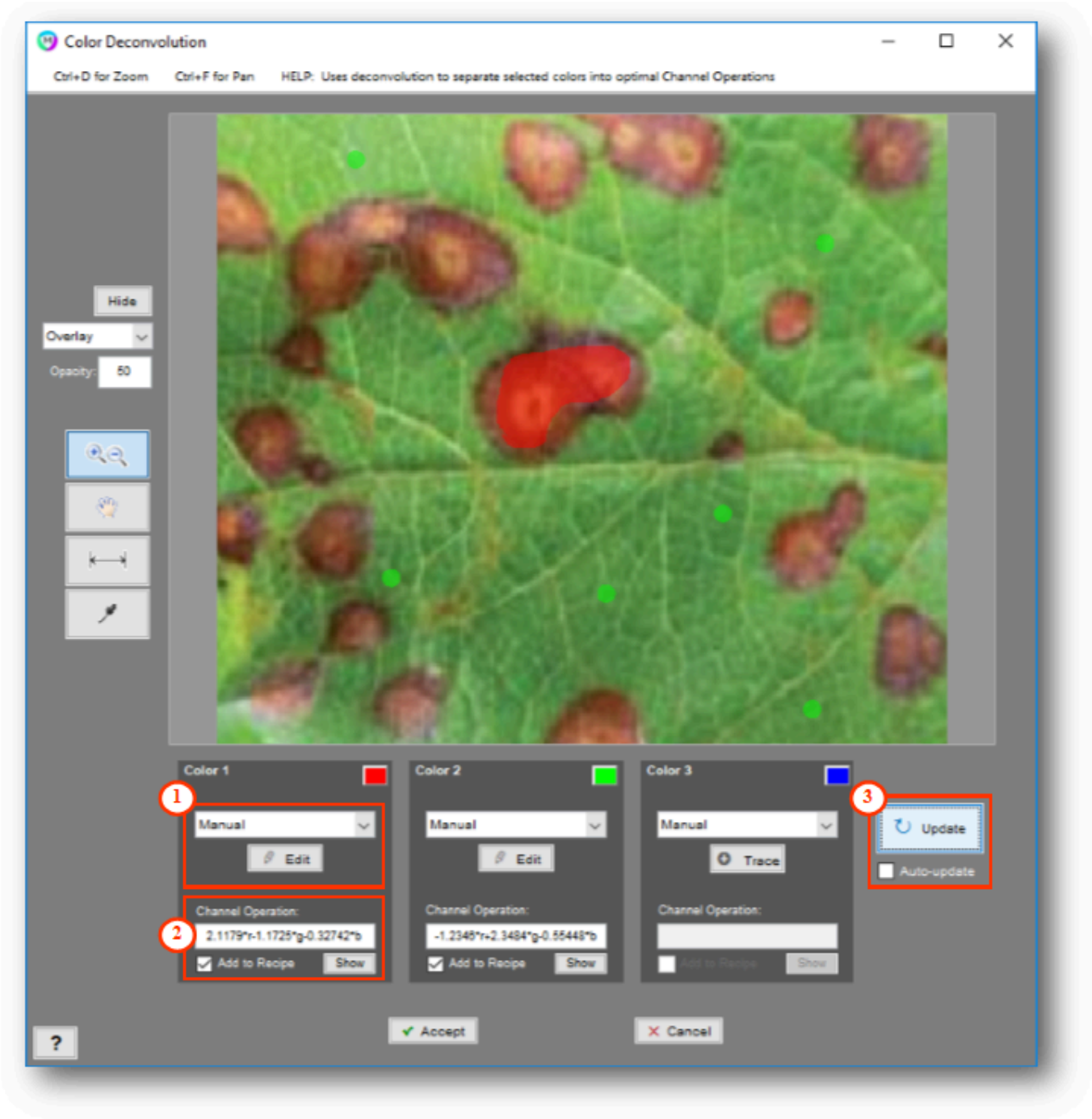
https://www.youtube.com/embed/xdrw2Utgb2c?rel=0

Color Deconvolution

Color > Color Deconvolution

Function is always applied to the original color image. Color Deconvolution is used to separate a color image into channels (colors) that are not the basic red, blue and green channels. It is a powerful tool in pre-processing color images that generates grayscale channel operations that can be further filtered to select features of interest.

Applications vary range from biomedical histology to mineral geology. Deconvolution works best on uncompressed file formats such as a TIFF.



1. Method

Select to manually outline the colors to be separated. Only option available at this time.

2. Resulting Channel Operation

Once the deconvolution is performed a channel operating is generated. Select Show to view the resulting grayscale image and check ‘Add to Recipe’ to automatically generated the channel operation in the Image Processor recipe.

3. Perform/Update

Once at least two of the color fields were selected, the image can be deconvoluted into channel operations which will appear under each color.

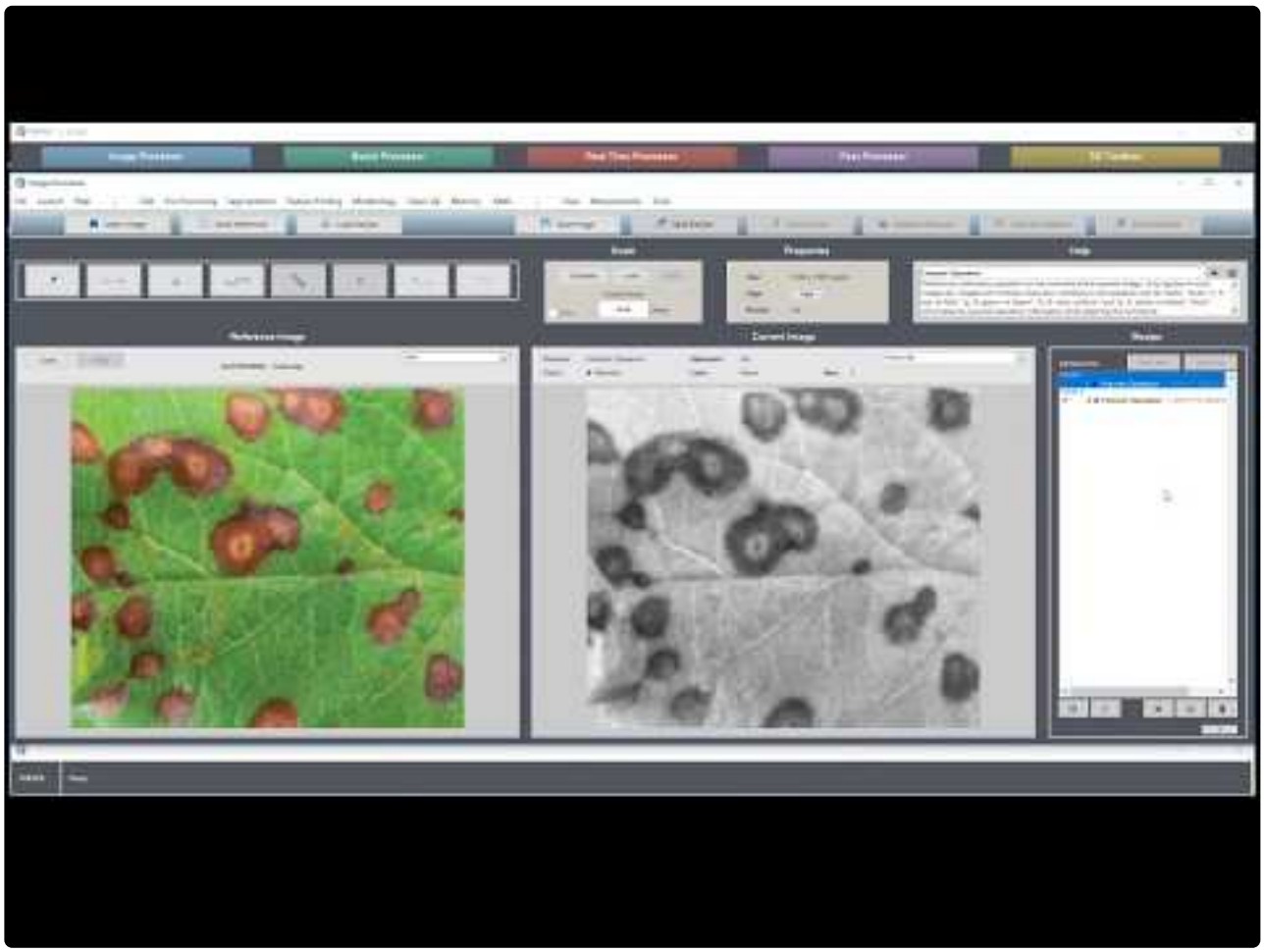
References

[1] Ruifrok, A. C., & Johnston, D. A. (2001). Quantification of histochemical staining by color deconvolution. Analytical and Quantitative Cytology and Histology, 23(4), 291–299. https://doi.org/10.1097/00129039-200303000-00014

Tips

- Color Deconvolution is always applied to the original color image
- Works best on uncompressed file formats such as TIFFs
- Critically important to be accurate and precise in manual selection of the colors

Tutorials



https://www.youtube.com/embed/KIkunloiWkE?rel=0

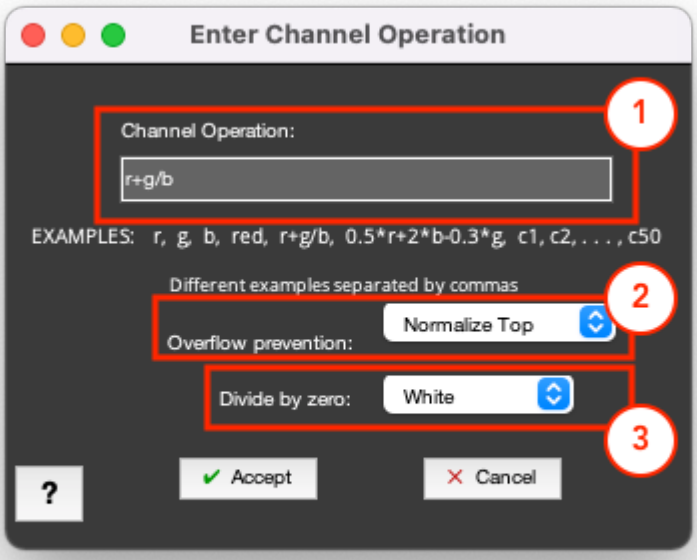
https://www.youtube.com/embed/KIkunloiWkE?rel=0

Channel Operation

Color > Channel Operation

Performs an arithmetic operation on the channels of the opened image. Only applies to color images (i.e. images with multiple channels).

Download a recipe of example channel operations for your color image. An example color image may be downloaded here.



1. Channel Operation

Variables in the operation can be “Auto”, “auto”; “r”, “R”, “red”, or “Red”; “g”, “G”, “green”, or “Green”; “b”, “B”, “blue” or “Blue”; “c1”, “c2”, . . . , “c50”; and “a”, “A”, “alpha”, or “Alpha”. “Auto” eliminates the hue and saturation information while retaining the luminance.

Mathematical operators can be “+”, “-”, “*”, “/”, “^”, “(” and “)”, “[” and “]”, and “~”.

2. Overflow

Specify post-processing to make sure that the results of the mathematical expression fit within the numerical limits of the image format (minimum is 0; maximum is 255 for 8-bit, 65,535 for 16-bit, and 1.0 for floating-point).

- **Normalize Top:** (default) The maximum pixel value of the expression's output is normalized to the format's max value, 0 remains at 0, and numbers below 0 are clipped to 0.
- **Normalize Bottom:** The minimum pixel value of the expression's output is normalized to 0, the format's max value remains at that max value, and numbers above the max value are clipped to the max value.
- **Normalize Both:** The maximum pixel value of the expression's output is normalized to the format's max value, and the minimum pixel value of the expression's output is normalized to 0.
- **Clip:** Numbers below 0 are set to 0, and numbers above the max value are set to that max value.

3. Divide by Zero

How to handle pixels which generate divide-by-zero events during evaluation of the mathematical expression:

- **White:** (default) X / 0 = white, -X / 0 = black, 0 / 0 = black
- **Black:** X / 0 = black, -X / 0 = white, 0 / 0 = black

Note that divide-by-zero pixels are ignored when calculating any normalization specified by the Overflow option.

Pre-Processing

Contains functions for manipulating grayscale pixel intensity. These functions as typically used as precursors to segmentation and clean-up steps, to improve the accuracy of the final feature selection.

Functions

Contrast

- Adjust Contrast
- Histogram Equalization
- *Histogram Match
- Flatten Background

Cluster

- Smart Cluster
- Superpixels

Noise Reduction

- Median Filter
- Wiener Filter
- Non-Local Means

Blur

- Gaussian Blur
- Average Blur
- Sum Filter
- Grayscale Dilate
- Grayscale Erode

Edges

- StdDev Filter
- Entropy Filter
- Gradient Filter
- Highlight Lines

Texture

- Bright Texture
- Dark Texture
- Advanced Texture
- Pattern Mapping
- Symmetry Mapping

- Orthogonal Correlate
- *Cross-Correlate

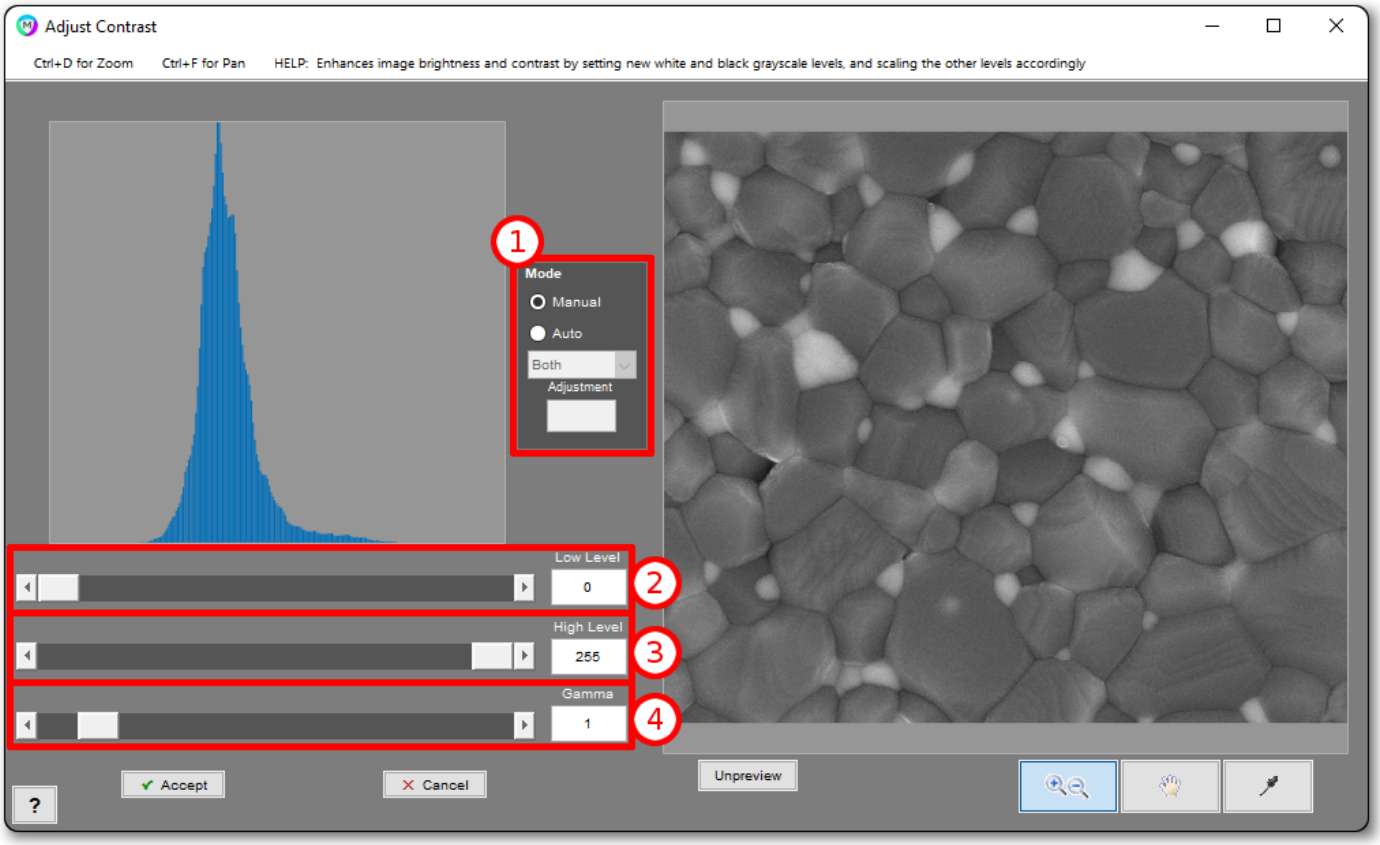
Correction

- Sharpen
- FFT Filter
- Make FFT
- Apply Stored FFT Filter
- *Grayscale Interpolation
- *Grayscale Reconstruction

Adjust Contrast

Pre-Processing > Adjust Contrast

Enhances image brightness and contrast by setting new white and black grayscale levels, and scaling the other levels accordingly. In 8-bit grayscale images, 0 is black and 255 is white by default. Image contrast can enhanced by redefining new pixel values to represent black and white, and then stretching the in-between values.



1. Mode

- **Manual:** Allows manual selection of levels

- **Auto:** Auto-selects new levels to maximize image contrast while minimizing information loss (levels are recalcualted to each image to which this step is applied)
  - **Drop Down:** Allows for automatic adjustment of either the Low Level, High Level or Both Levels.
  - **Adjustment:** Specifies offset to make to auto-determined levels. A positive adjustment spreads the auto-determined levels, and a negative adjustment contracts them.

2. Low Level

Pixel value to be the new black level

3. High Level

Pixel value to be the new white level

4. Gamma

Controls the linearity of the black to white gradient of the grayscale spectrum. Lower values will more favor the white end, and higher values the black. 1 is a perfect linear gradient.

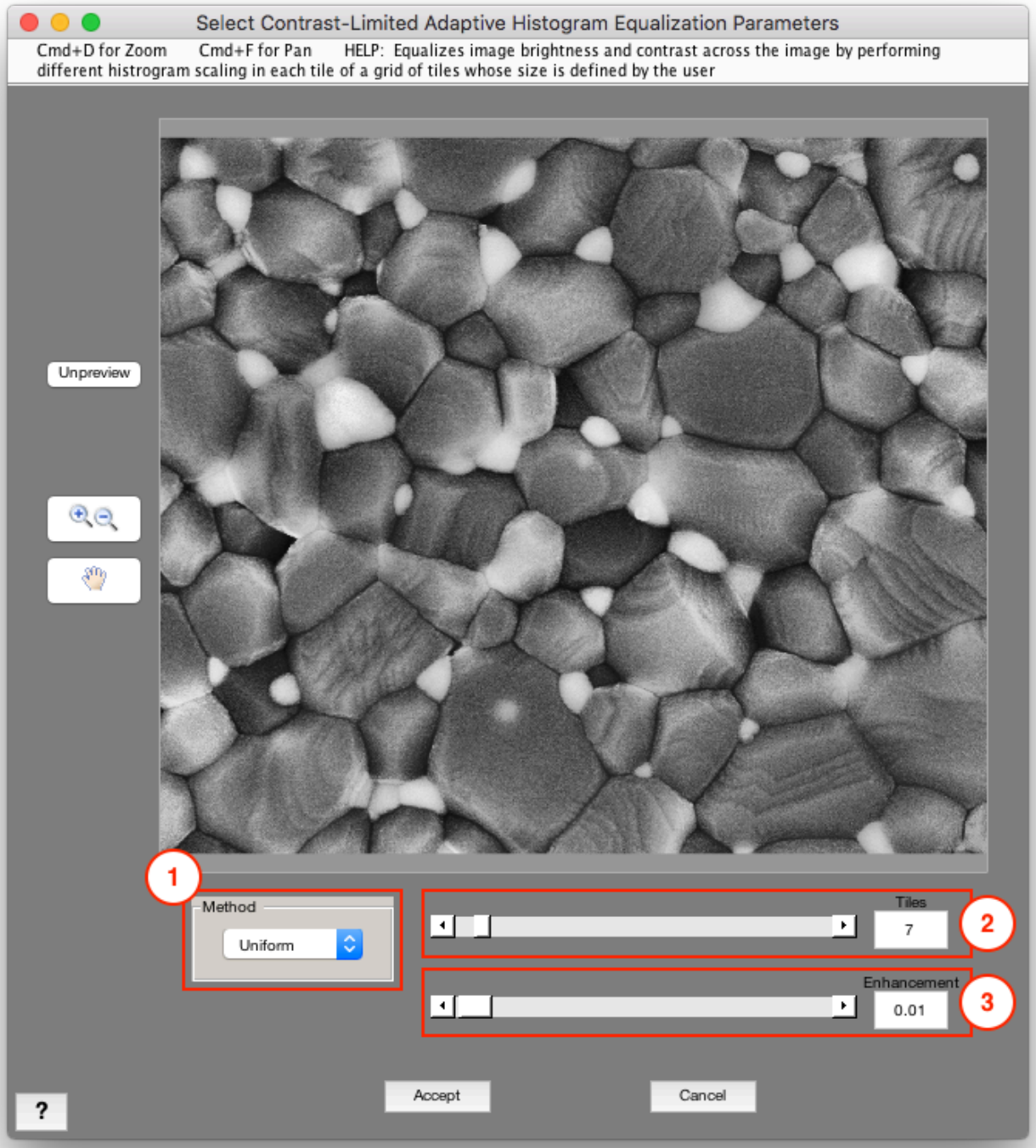
Tips

Use Auto>High to white balance your image and use Auto>Low to black balance your image.

Histogram Equalization

Pre-Processing > Histogram Equalization

Equalizes image brightness and contrast across the image by performing different histogram scaling in each tile of grid of tiles whose size is defined by user [1].



1. Method

Controls desired histogram shape for the image tiles

- **Uniform:** Flat-histogram
- **Rayleigh:** Bell-shaped histogram
- **Exponential:** Curved histogram

2. Tiles

Number of tiles to split the image into in each direction (Recommended: 8)

3. Enhancement

Adjusts amount of contrast enhancement. Higher values result in more contrast (Recommended: 0.01)

References

[1] Zuiderveld, Karel. “Contrast Limited Adaptive Histograph Equalization.” Graphic Gems IV. San Diego: Academic Press Professional, 1994. 474–485.

*Histogram Match

Pre-Processing > *Histogram Match

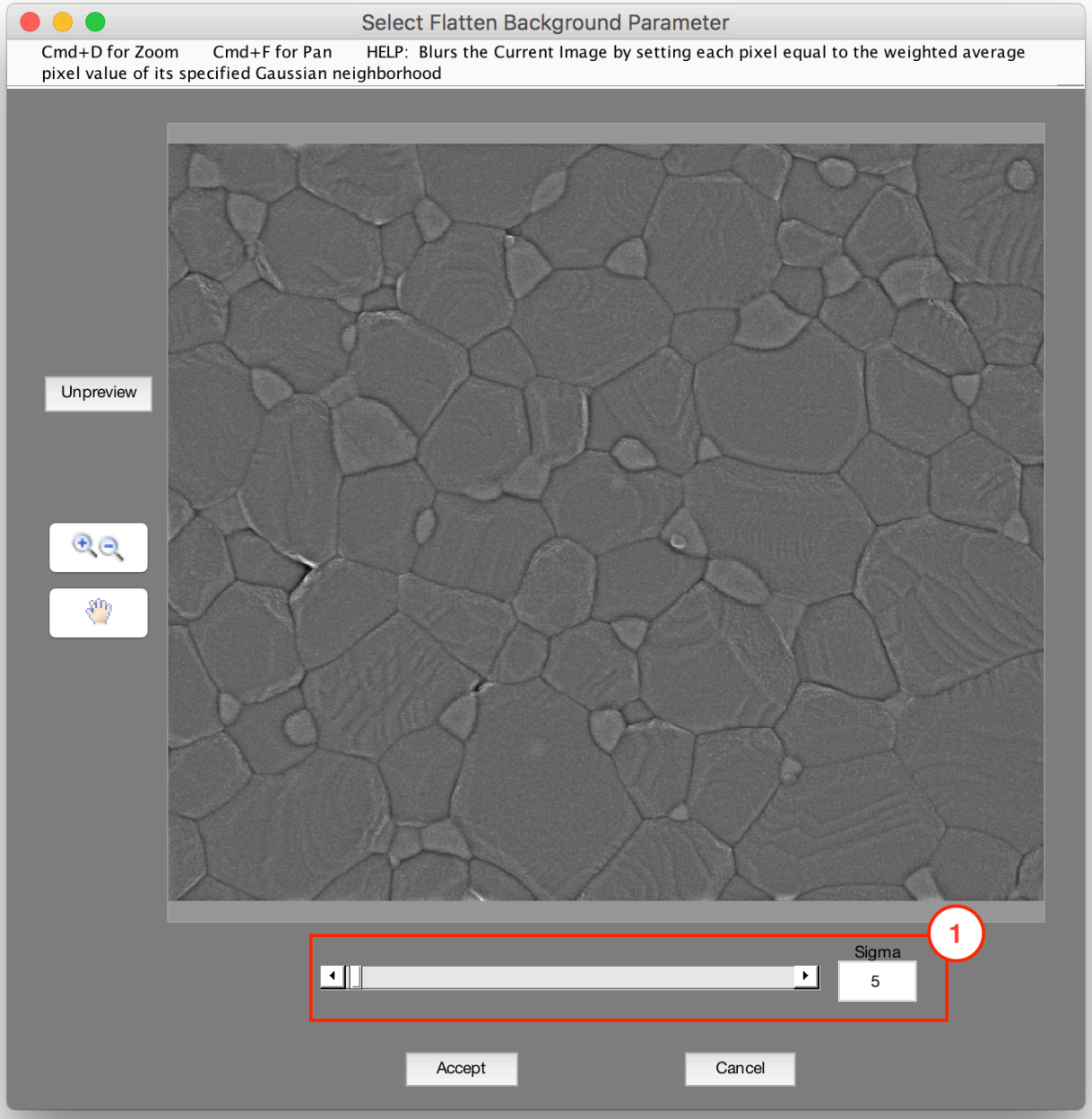
Requires Companion Image

Modifies the overall contrast of the Current Image to match as closely as possible to the overall contrast of the Companion Image. Often used when you have a series of images that you want to normalize the contrast of. In this case use Memory > Load Companion Image to load the “comparison image” into the Companion Image slot. Then use Memory > Call Original Image to bring forward the original Opened Image. Then Histogram Match will adjust the contrast of the Opened Image to match that of the Companion Image (i.e., “comparison image”).

Flatten Background

Pre-Processing > Flatten Background

Helps remove brightness gradients by applying a Gaussian blur to the Current Image and then subtracting the blurred image from the Current Image.



1. Sigma

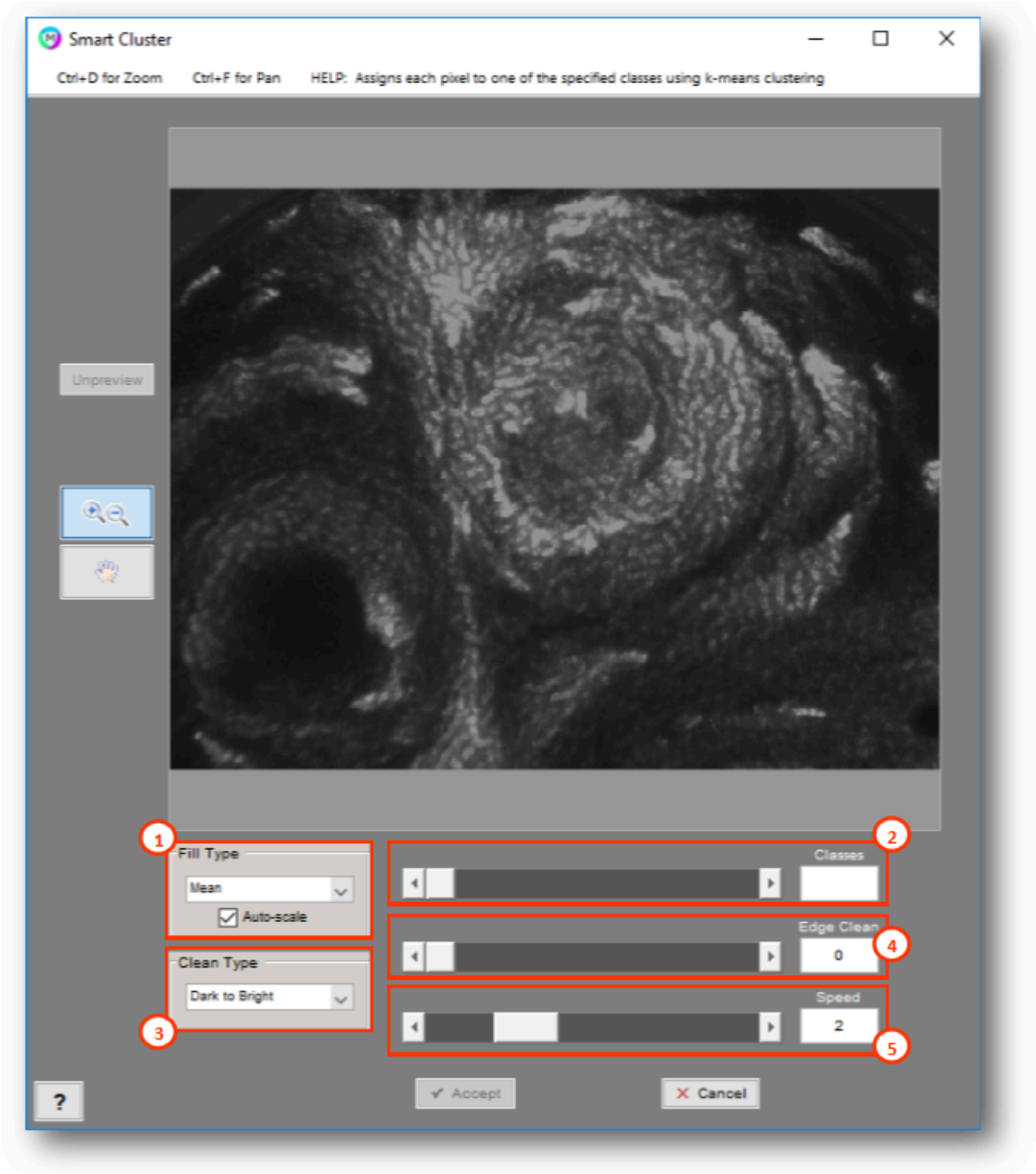
Size of neighborhood (in pixels) which is considered about each pixel for the filter

Smart Cluster

Segmentation > Smart Cluster

Assigns each pixel to one of the specified classes using k-means clustering [1-4]. This can be a powerful and objective way of segmenting images. Works especially well for images with 3 or more feature types of interest, where different grayscale levels define the different feature types. This function results in a grayscale image where each class is colored according to the specified statistic of the underlying pixels.

To split the resulting image into different Layers, add a Set Memory image step afterwards, then iteratively call that restore image and use “Range Threshold” steps to select the different classes as B/W images.



1. Fill Type

Statistic to use to fill each cluster. Check “Auto Scale” to set the darkest class to black, set the brightest class to white, and scale in-between classes accordingly.

- **Mean:** Fills each cluster with mean value of underlying pixels

- **Median:** Fills each cluster with median value of underlying pixels
- **StdDev:** Fills each cluster with standard deviation of underlying pixels
- **Min:** Fills each cluster with minimum value of underlying pixels
- **Max:** Fills each cluster with maximum value of underlying pixels
- **Class:** Fills each cluster with value of its class. Image is normalized from 0 to 255 afterwards for easier visualization.

2. Classes

Number of classes to cluster each pixel into. If you are trying to segment one set of features from the background, you should enter 2 classes.

3. Edge Type

Class order for edge cleaning

- **Dark to Bright:** Expands each class to clean edges from darkest class to brightest class. Recommended to clean edge artifacts for dark features on bright backgrounds.
- **Bright to Dark:** Expands each class to clean edges from brightest class to darkest class. Recommended to clean edge artifacts for bright features on dark backgrounds.

4. Edge Clean

Factor which controls extent to which edge artifacts are removed. The value indicates the max pixel thickness of edge artifacts that will be removed. Use 0 for no cleaning.

5. Speed

Parameter that controls the resolution of the cluster. Lower speed results in higher resolution clustering, but takes longer. Higher speed results in faster execution but lower resolution clustering.

References

[1] Arthur, David, and Sergi Vassilvitskii. “K-means++: The Advantages of Careful Seeding.” SODA ‘07: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms. 2007, pp. 1027–1035.

[2] Lloyd, Stuart P. “Least Squares Quantization in PCM.” IEEE Transactions on Information Theory. Vol. 28, 1982, pp. 129–137.

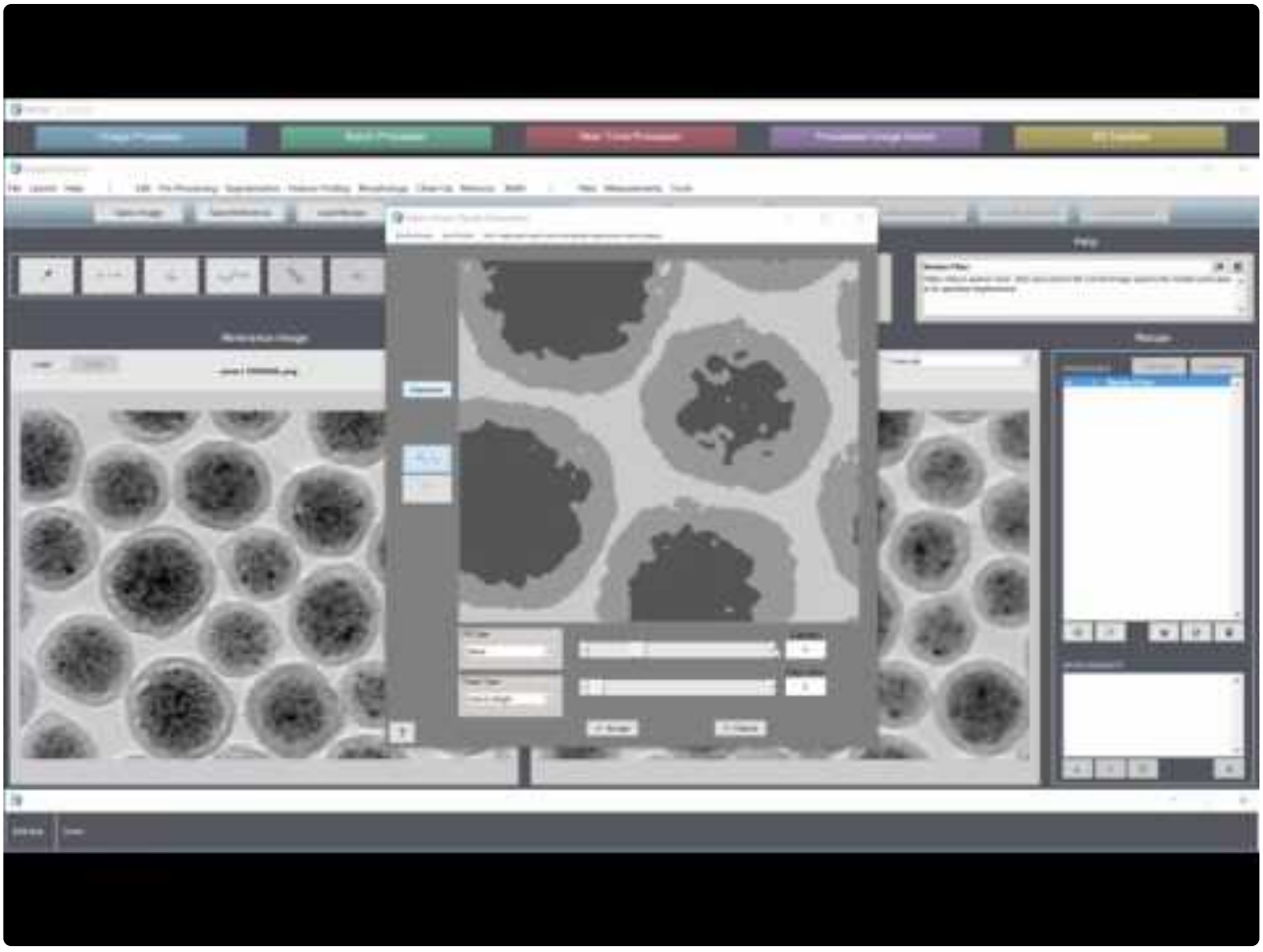
[3] Seber, G. A. F. Multivariate Observations. Hoboken, NJ: John Wiley & Sons, Inc., 1984.

[4] Spath, H. Cluster Dissection and Analysis: Theory, FORTRAN Programs, Examples. Translated by J. Goldschmidt. New York: Halsted Press, 1985.

Tips

- Smart cluster with 2 classes is equivalent to a basic threshold with the ‘Auto’ setting, avoid this to improve objectivity
- Use smart cluster to objectively classify grayscale image data into discrete feature clusters where the number of classes is equal to the number of discrete features you’re interested in.

Tutorial



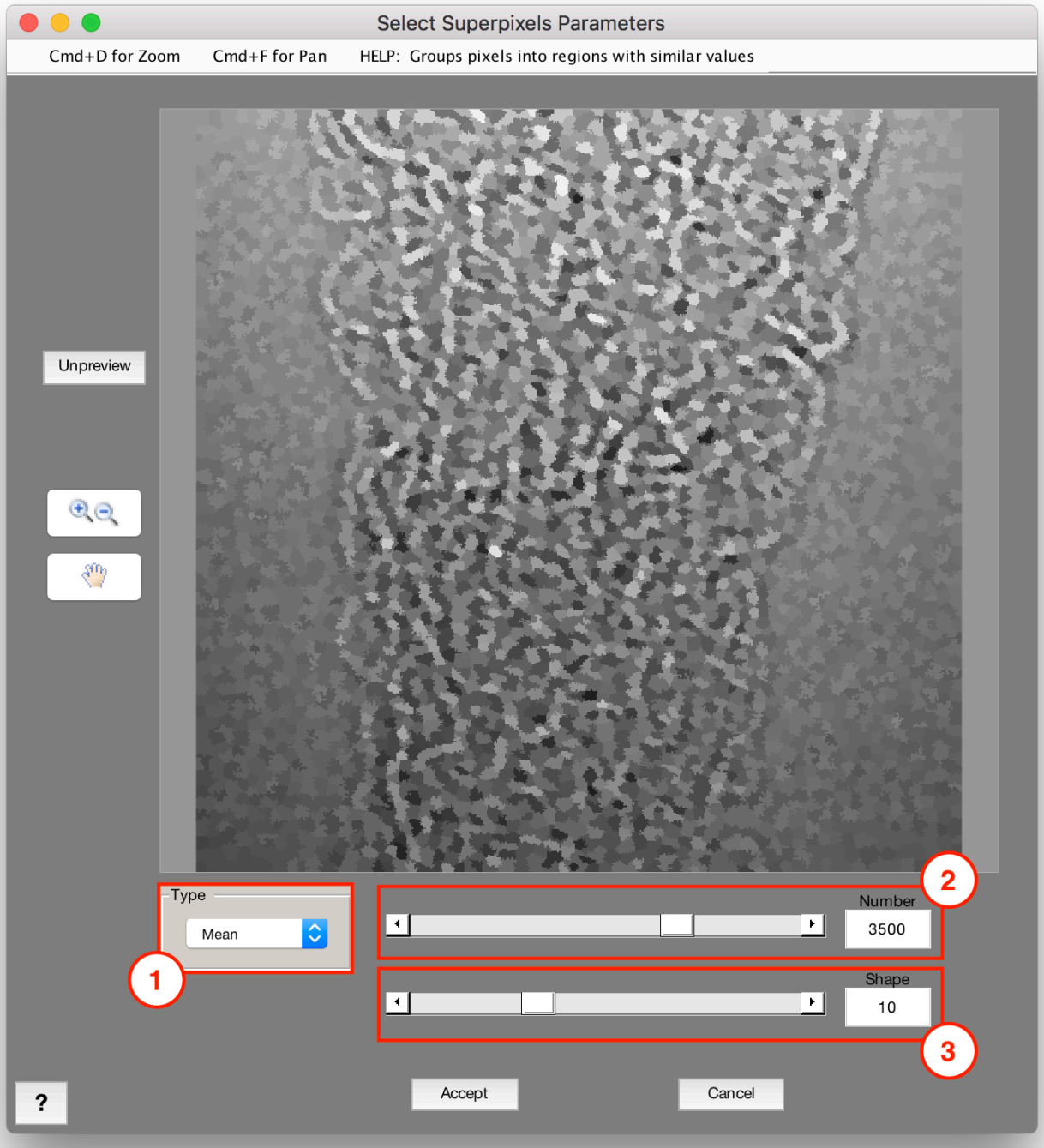
https://www.youtube.com/embed/FxebQbpdRuE?rel=0

https://www.youtube.com/embed/FxebQbpdRuE?rel=0

Superpixels

Segmentation > Superpixels

Groups pixels into regions with similar values. This can be a helpful processing step prior to other segmentation or feature-finding steps. Uses the simple linear iterative clustering (SLIC) algorithm [1,2].



1. Type

Statistic to use to fill each cluster

- **Mean:** Fills each superpixel with mean value of underlying pixels
- **Median:** Fills each superpixel with median value of underlying pixels
- **Mode:** Fills each superpixel with mode value of underlying pixels
- **StdDev:** Fills each superpixel with standard deviation of underlying pixels
- **Min:** Fills each superpixel with minimum value of underlying pixels
- **Max:** Fills each superpixel with maximum value of underlying pixels

2. Number

The target number of superpixels to create. Actual number may be different.

3. Shape

Factor which controls the shape of the superpixels. Lower numbers produce more realistic shapes that follow boundaries. (Recommended: 10)

References

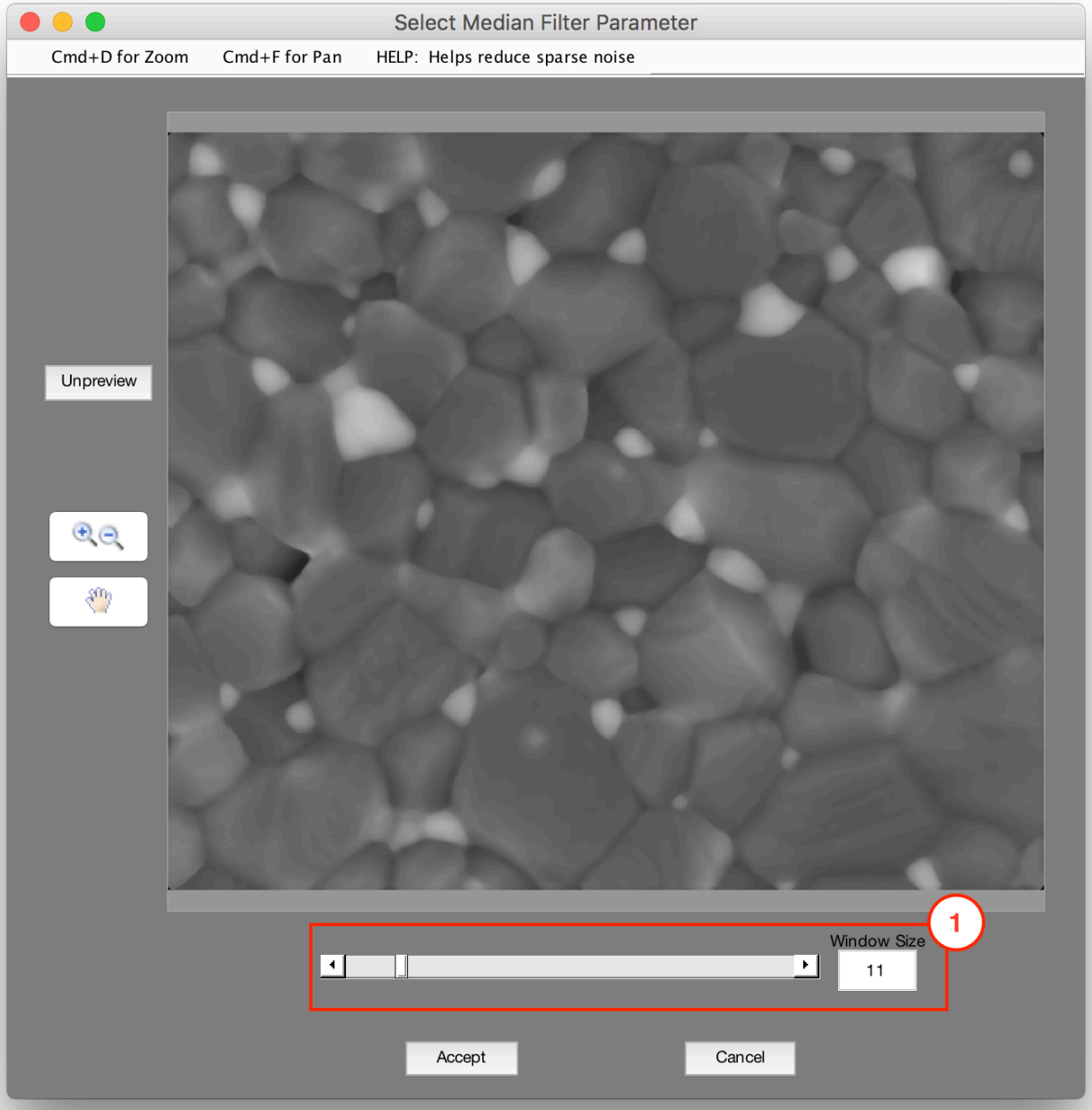
[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Susstrunk, SLIC Superpixels Compared to State-of-the-art Superpixel Methods. IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 34, Issue 11, pp. 2274-2282, May 2012

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Susstrunk, SLIC Superpixels. EPFL Technical Report, no. 149300, June 2010.

Median Filter

Pre-Processing > Median Filter

Helps reduce sparse noise. Sets each pixel in the Current Image equal to the median pixel value of its specified neighborhood.



1. Window Size

Size of neighborhood (in pixels) which is considered about each pixel for the filter

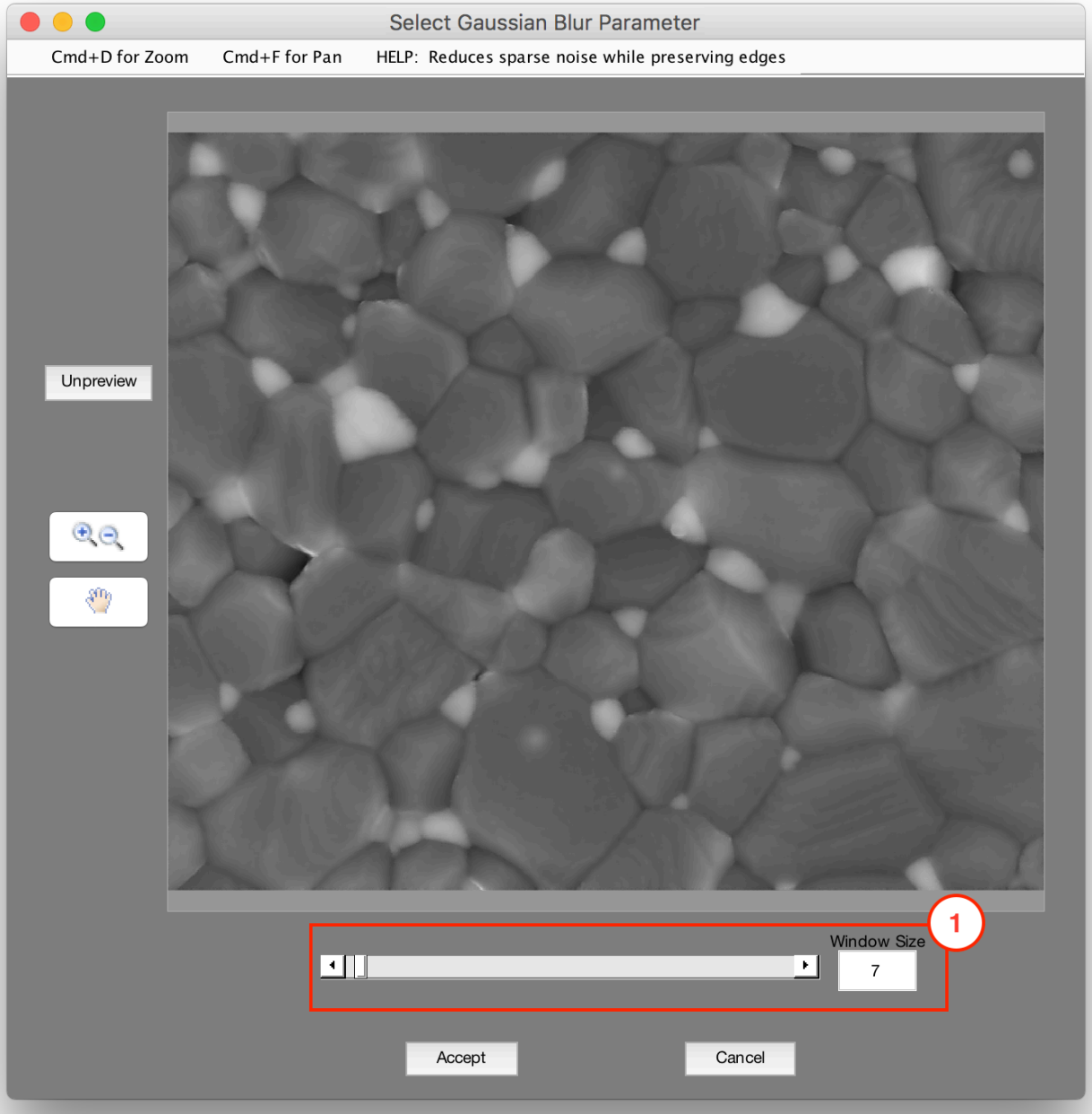
Tips

- Noise-reduction filters are often very useful for removing granular noise while maintaining clean feature boundaries.
- We recommend you try all of the filters to find which once works best.
- Starting Window Size Value: 7, increase to remove larger artifacts.

Wiener Filter

Pre-Processing > Wiener Filter

Reduces sparse noise while preserving edges. Sets each pixel in the Current Image equal to the median pixel value of its adjusted specified neighborhood size [1]. Pixels with near edges will use smaller neighborhood sizes.



1. Window Size

Size of neighborhood (in pixels) which is considered about each pixel for the filter

References

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

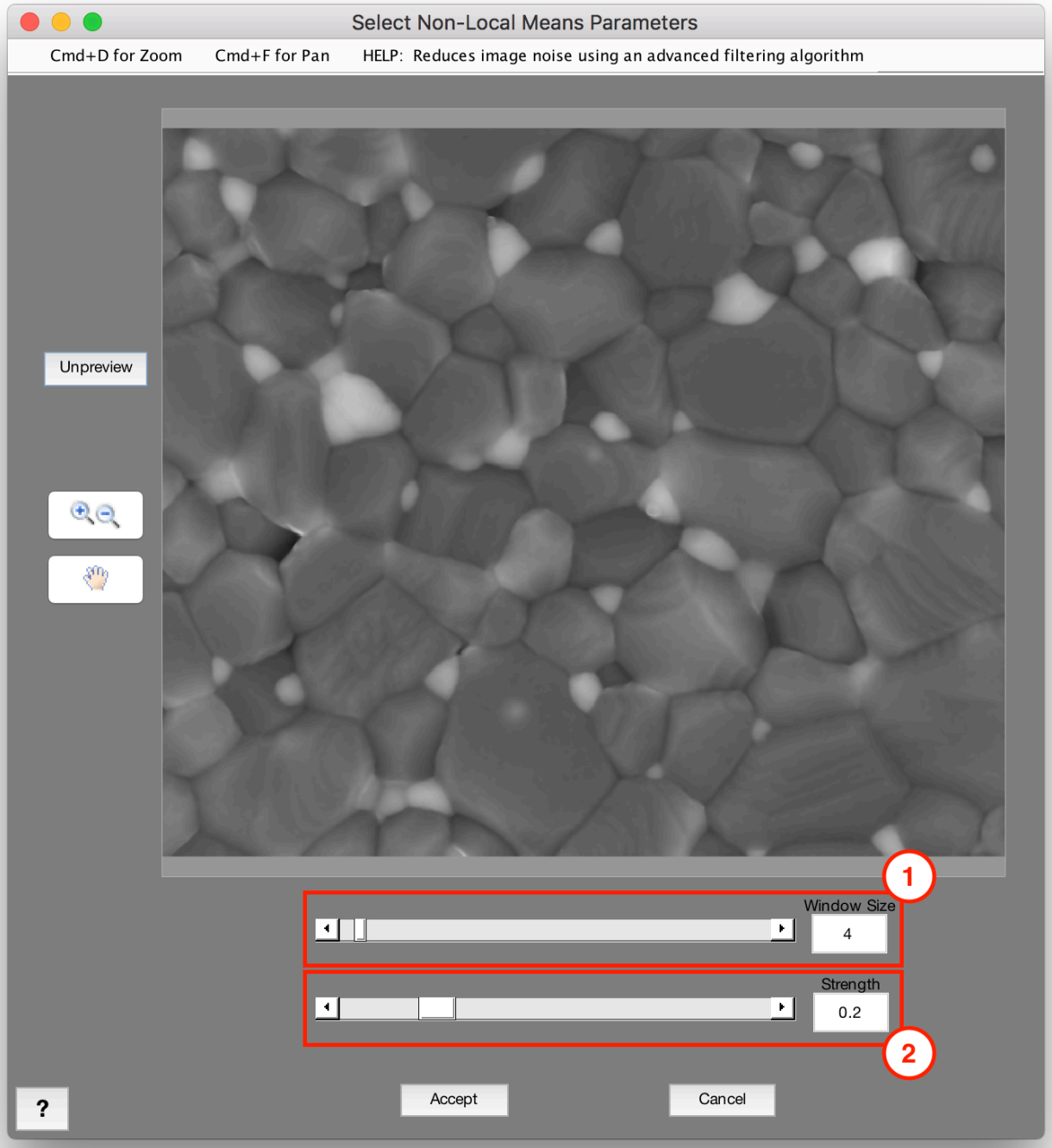
Tips

- Noise-reduction filters are often very useful for removing granular noise while maintaining clean feature boundaries.
- We recommend you try all of the filters to find which once works best.
- Starting Window Size Value: 5, increase to remove larger artifacts

Non-Local Means

Pre-Processing > Non-Local Means

Reduces image noise using an advanced filtering algorithm [1]. Each pixel''s local window is compared to windows around it. The windows' center pixels are averaged together with weights depending on the variations between the windows.



1. Window Size

Size of neighborhood (in pixels) which is considered about each pixel for the filter **(RECOMMENDED: 2-6)**

2. Strength

Factor which controls the strength of the filter (Recommended: 0.2-0.6)

References

[1] Buades, A., B. Coll, and J.-M. Morel. “A Non-Local Algorithm for Image Denoising.” 2005 IEEE® Computer Society Conference on Computer Vision and Pattern Recognition. Vol. 2, June 2005, pp. 60–65.

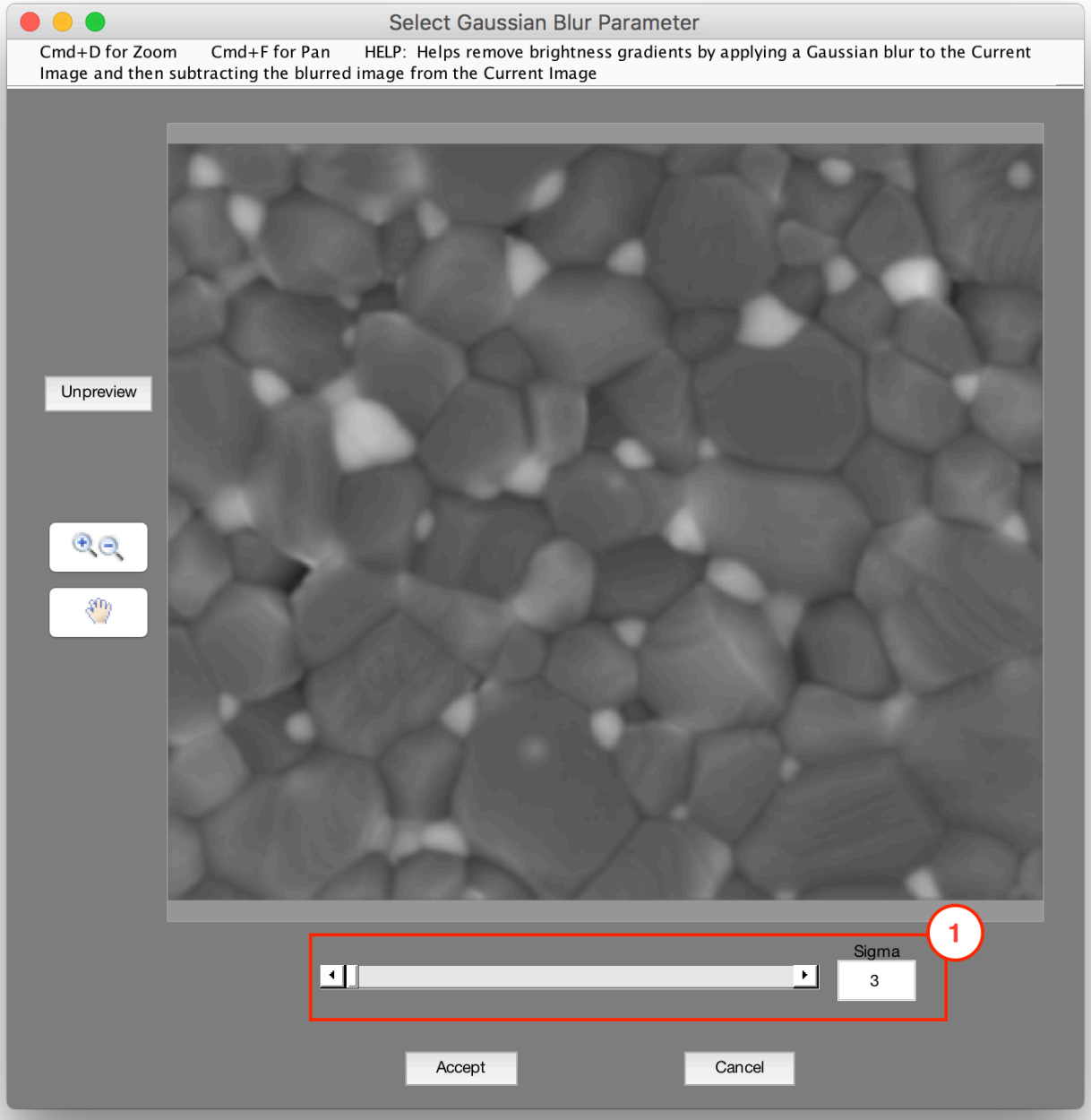
Tips

- Noise-reduction filters are often very useful for removing granular noise while maintaining clean feature boundaries.
- We recommend you try all of the filters to find which once works best.
- Non-Local Means is one of the more powerful pre-processing filters MIPAR has to offer, but can take longer to execute for larger images.
- Starting Window Size Value: 2-4, Strength: 0.2, decrease Strength for low resolution images. Increasing window size above 20 increases time of execution with little changes.

Gaussian Blur

Pre-Processing > Gaussian Blur

Blurs the Current Image by setting each pixel equal to the weighted average pixel value of its specified Gaussian neighborhood.



1. Sigma

Size of neighborhood (in pixels) which is considered about each pixel for the filter

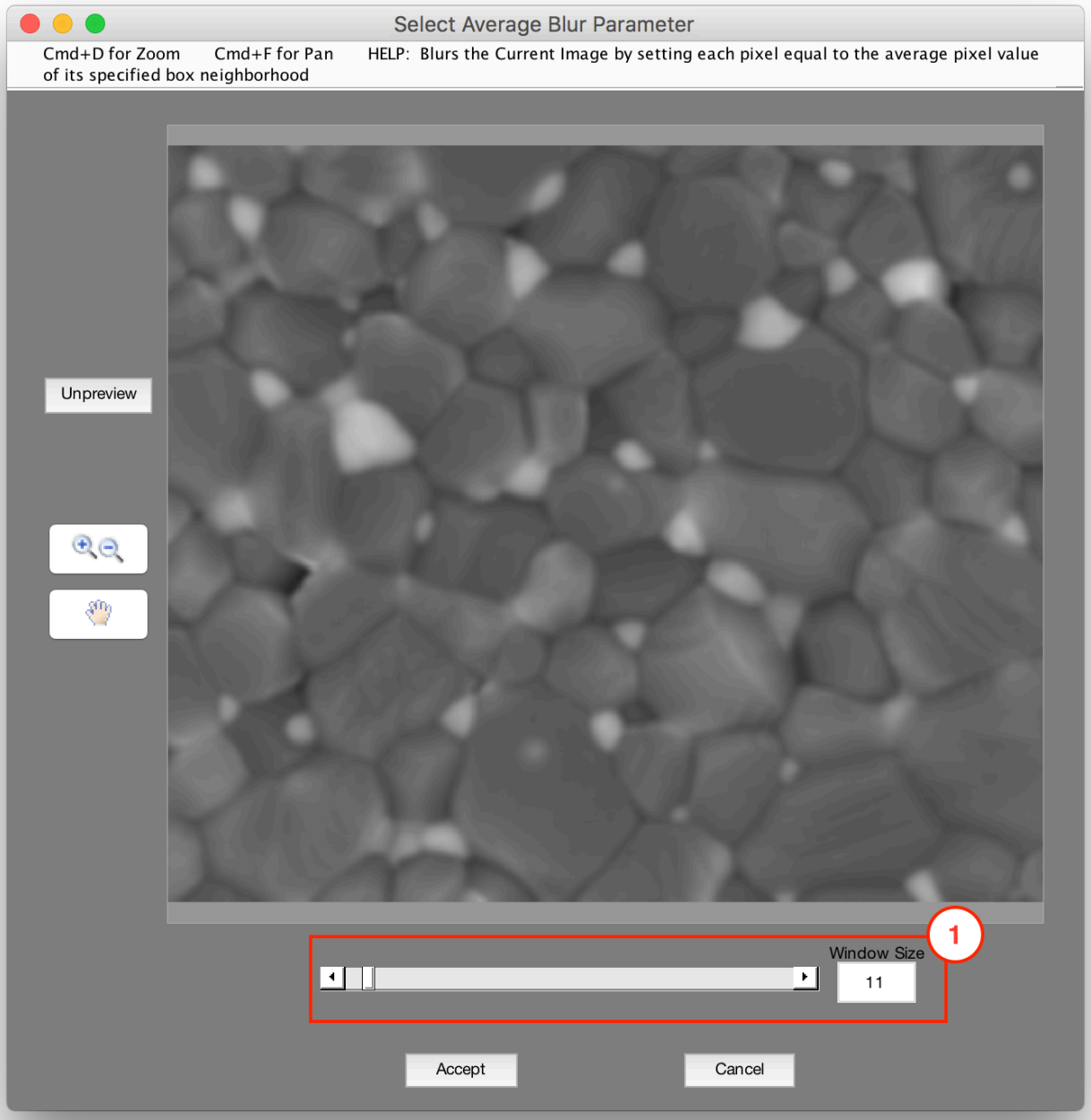
Tips

Blur filters are often very useful for removing small artifacts in order to threshold larger features.

Average Blur

Pre-Processing > Average Blur

Blurs the Current Image by setting each pixel equal to the average pixel value of its specified box neighborhood.



1. Window Size

Size of neighborhood (in pixels) which is considered about each pixel for the filter

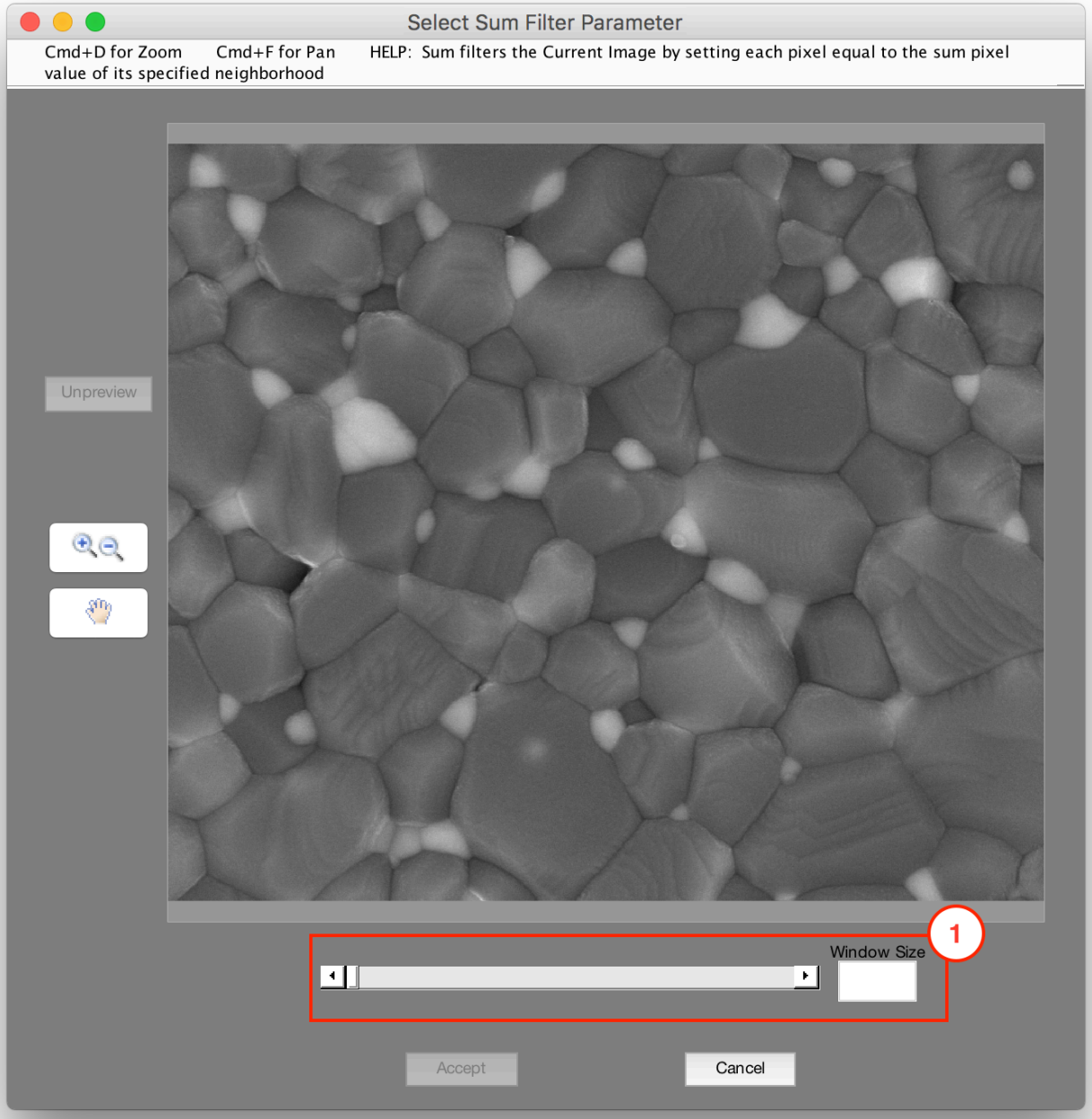
Tips

Blur filters are often very useful for removing small artifacts in order to threshold larger features.

Sum Filter

Pre-Processing > Sum Filter

Sum filters the Current Image by setting each pixel equal to the sum pixel value of its specified neighborhood.



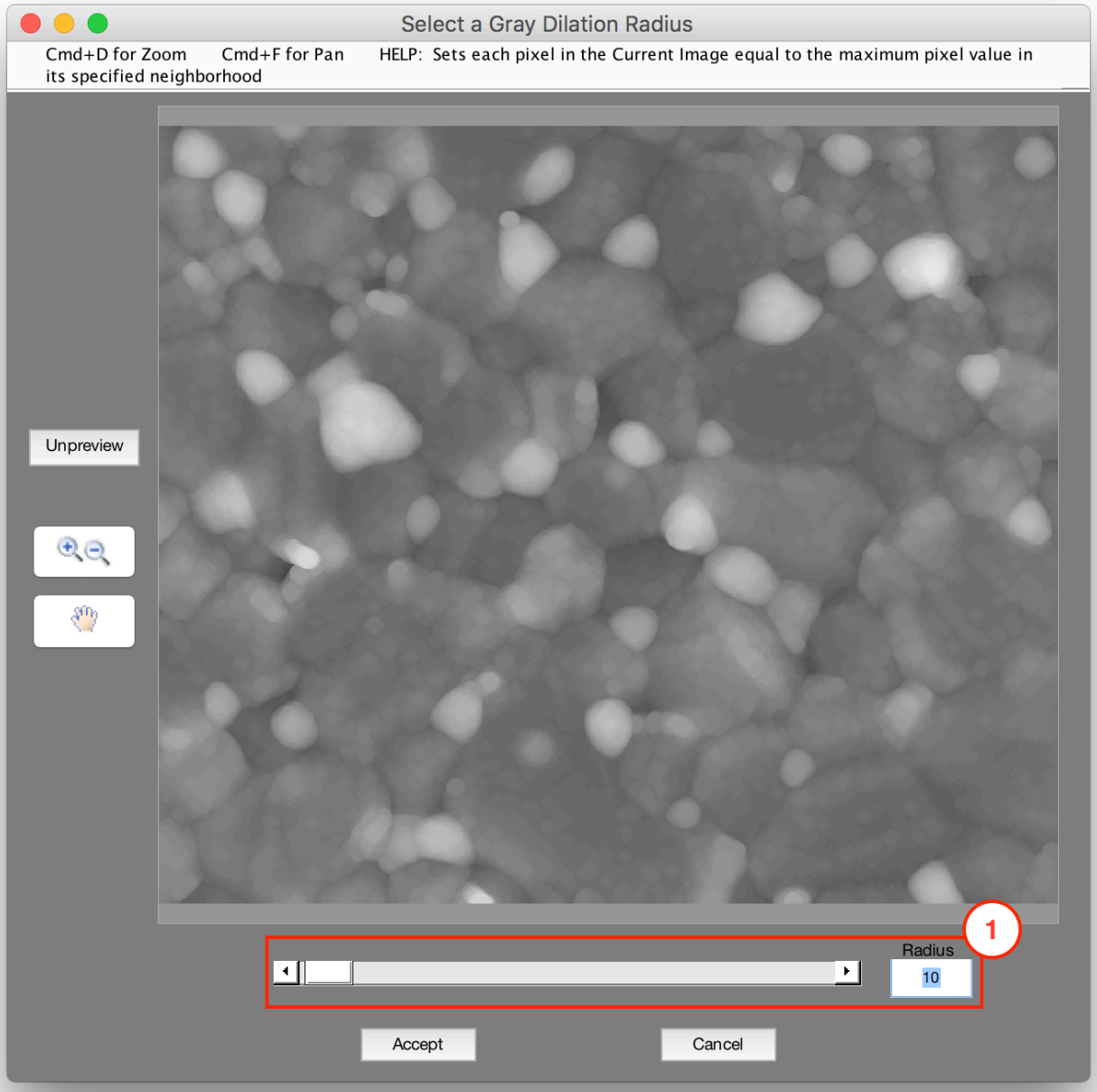
1. Window Size

Size of neighborhood (in pixels) which is considered about each pixel for the filter

Grayscale Dilate

Pre-Processing > Grayscale Dilate

Sets each pixel in the Current Image equal to the maximum pixel value in its specified neighborhood.



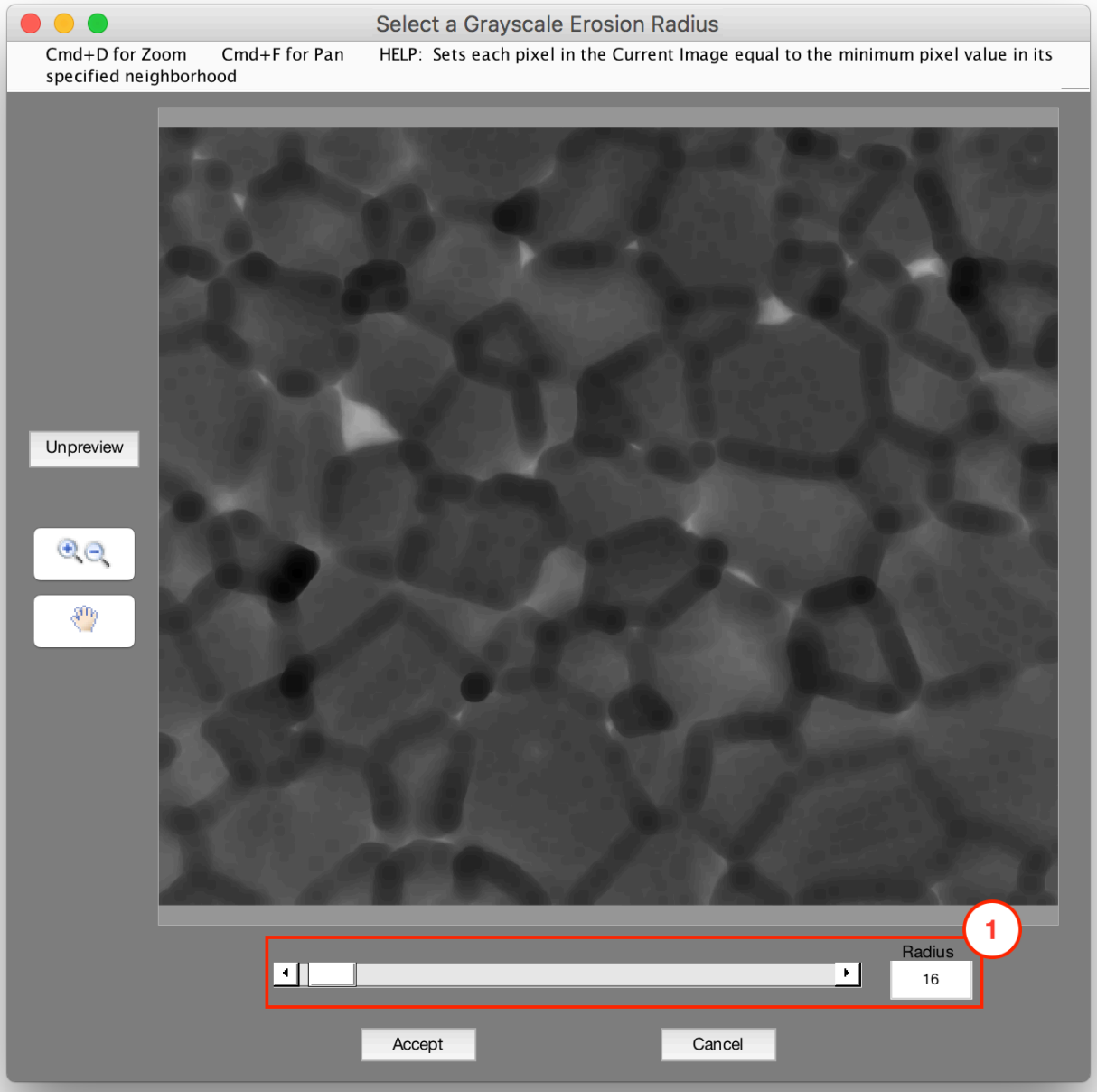
1. Radius

Radius (in pixels) to consider about each pixel to calculate the local maxima

Grayscale Erode

Pre-Processing > Grayscale Erode

Sets each pixel in the Current Image equal to the minimum pixel value in its specified neighborhood.



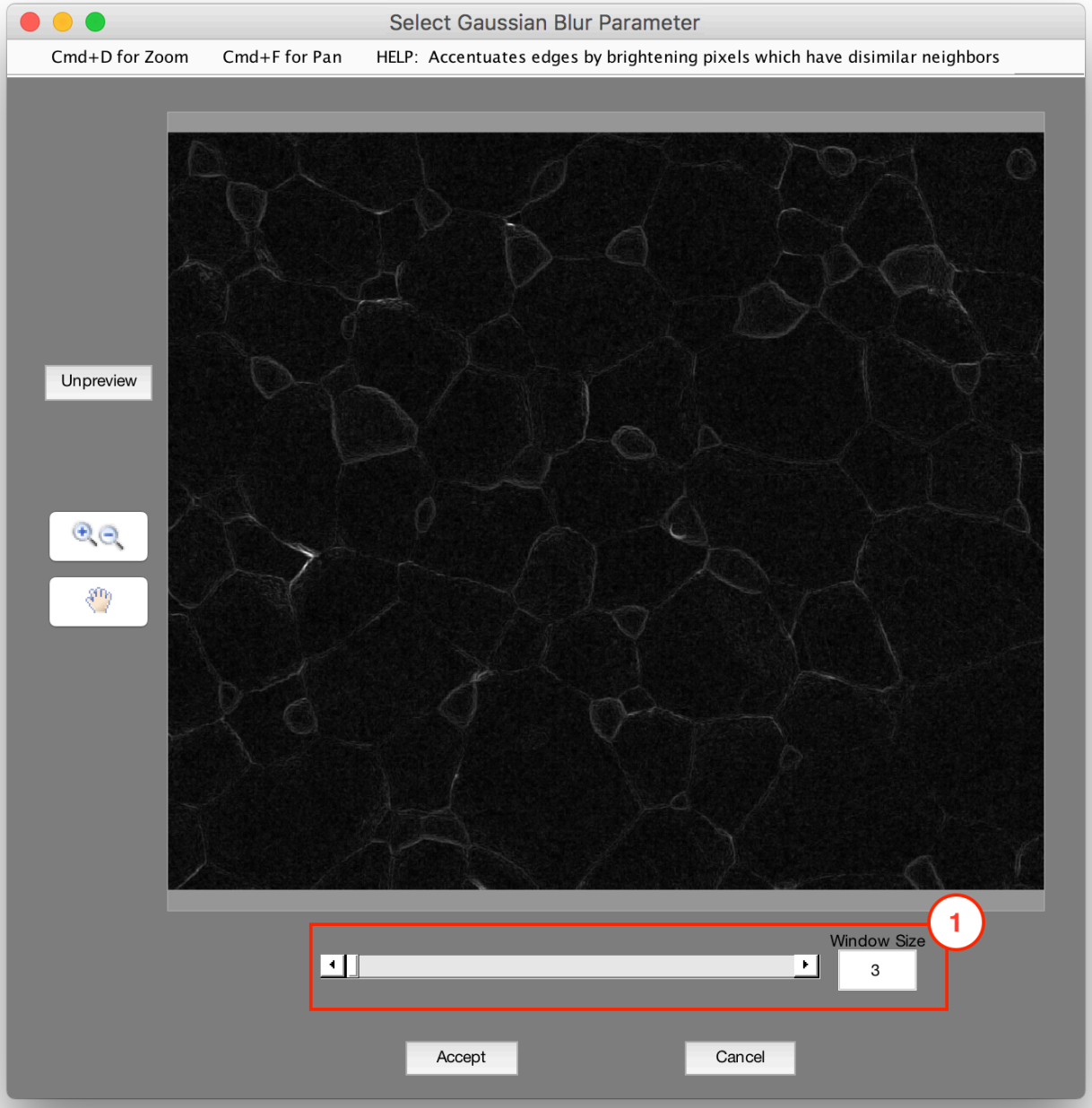
1. Radius

Radius (in pixels) to consider about each pixel to calculate the local minima

StdDev Filter

Pre-Processing > StdDev Filter

Accentuates edges by brightening pixels which have dissimilar neighbors. Sets each pixel in the Current Image equal to the standard deviation of the pixel values in its specified neighborhood.



1. Window Size

Size of neighborhood (in pixels) which is considered about each pixel for the filter

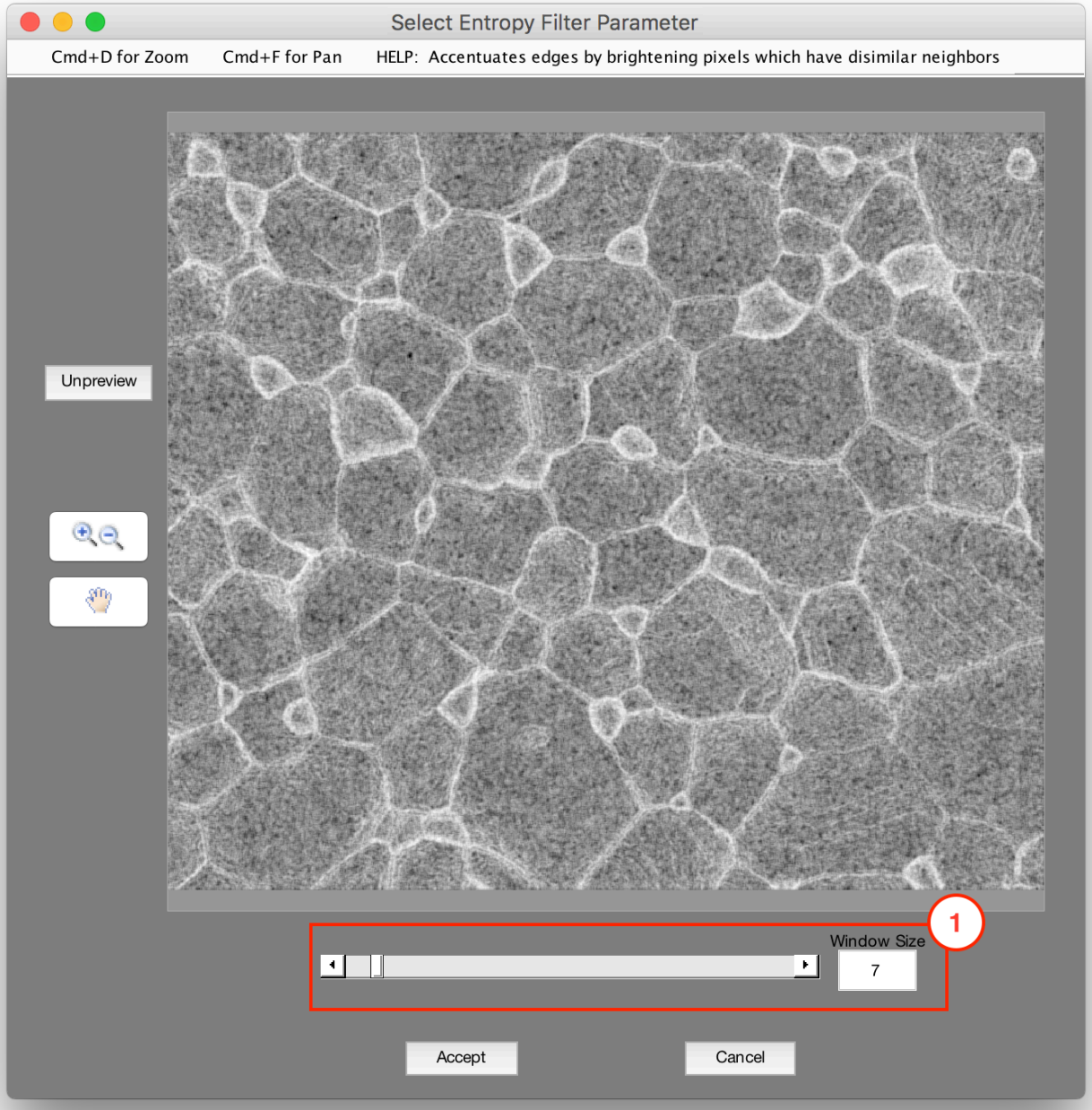
Tips

- Used to find feature boundaries and to increase the contrast of features differing in intensity.
- Powerful filter that can be used in combination with Find Edges, Dilation, Erosion and the Optimization Engine to improve segmentation of features.
- Example 1 (Faint grain boundaries): StdDev Filter
- Example 2 (Subjective feature Dilation/Erosion): StdDev Filter
- Starting Window Size: 5, increase to overpower small artifacts, decrease to maintain an accurate feature edge

Entropy Filter

Pre-Processing > Entropy Filter

Accentuates edges by brightening pixels which have dissimilar neighbors. Sets each pixel in the Current Image equal to the entropy (measure of disorder) of the pixel values in its specified neighborhood [1].



1. Window Size

Size of neighborhood (in pixels) which is considered about each pixel for the filter

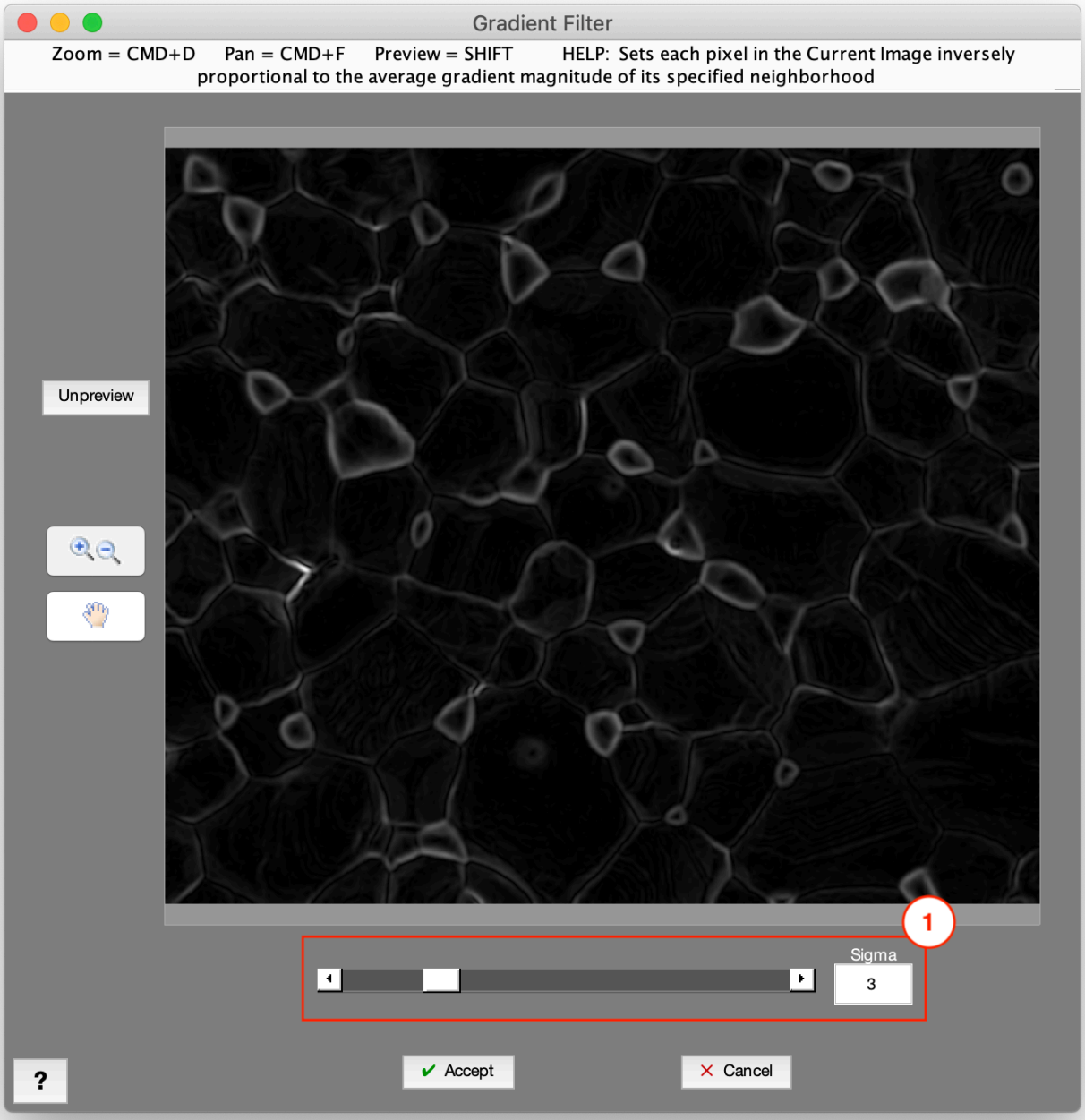
References

[1] Gonzalez, R.C., R.E. Woods, S.L. Eddins, Digital Image Processing Using MATLAB, New Jersey, Prentice Hall, 2003, Chapter 11.

Gradient Filter

Pre-Processing > Gradient Filter

Sets each pixel in the Current Image inversely proportional to the average gradient magnitude of its specified neighborhood.



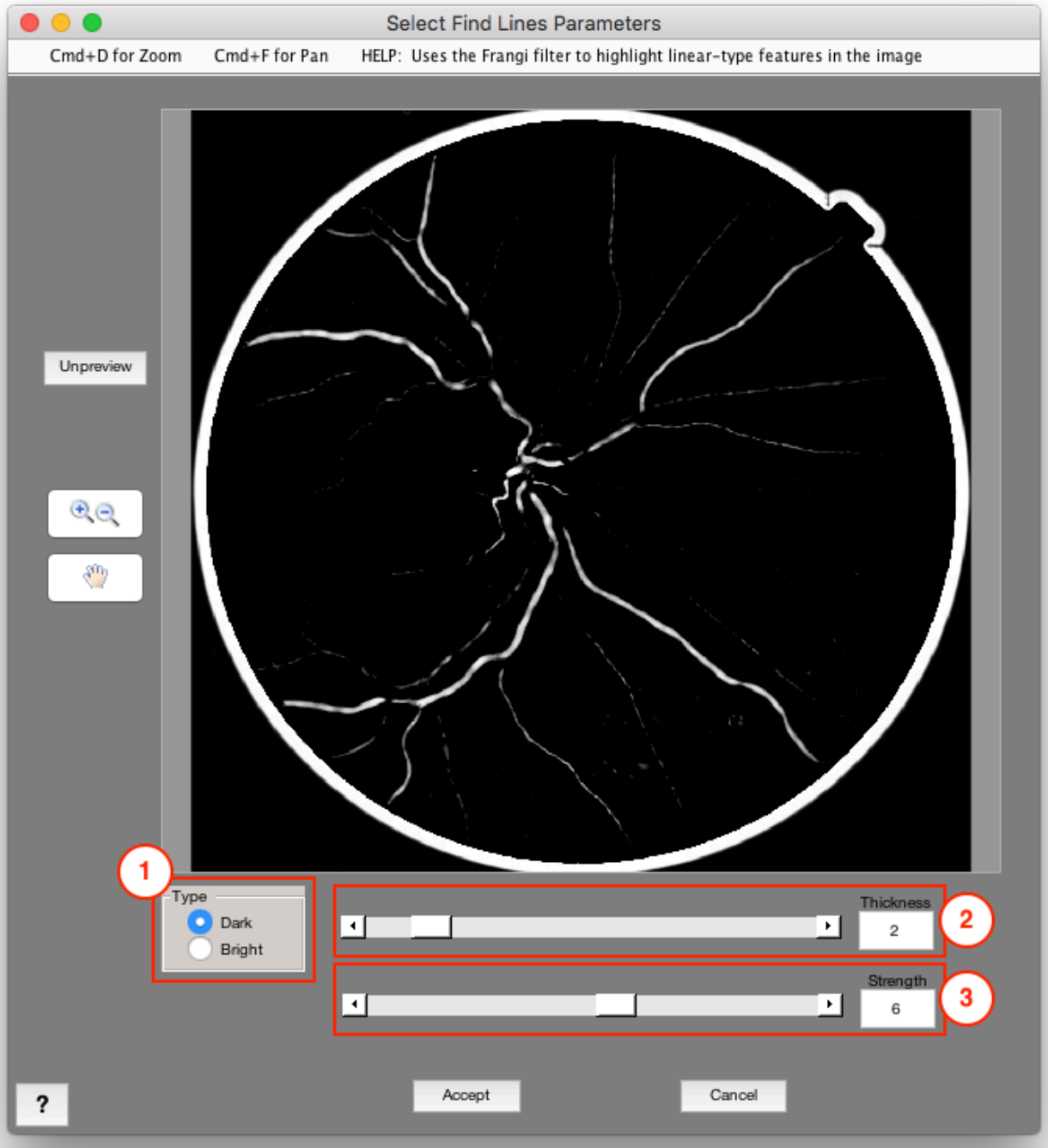
1. Sigma

Size of neighborhood (in pixels) which is considered about each pixel for the filter

Highlight Lines

Pre-Processing > Highlight Lines

Uses the Frangi filter [1] to highlight linear-type features in the image. Combine with a threshold afterwards to select certain linear-type features.



1. Type

- **Dark:** Highlight dark linear-type features
- **Bright:** Highlight bright linear-type features

2. Thickness

Factor which controls the thickness of the lines that are targeted (Recommended: 1-3)

3. Strength

Factor which controls the strength of the Frangi (line-highlighting) filter (Recommended: 1-3)

Tips

- Useful for identifying continuous lines in an image. Image pre-processing is recommended to reduce noise when using this function.

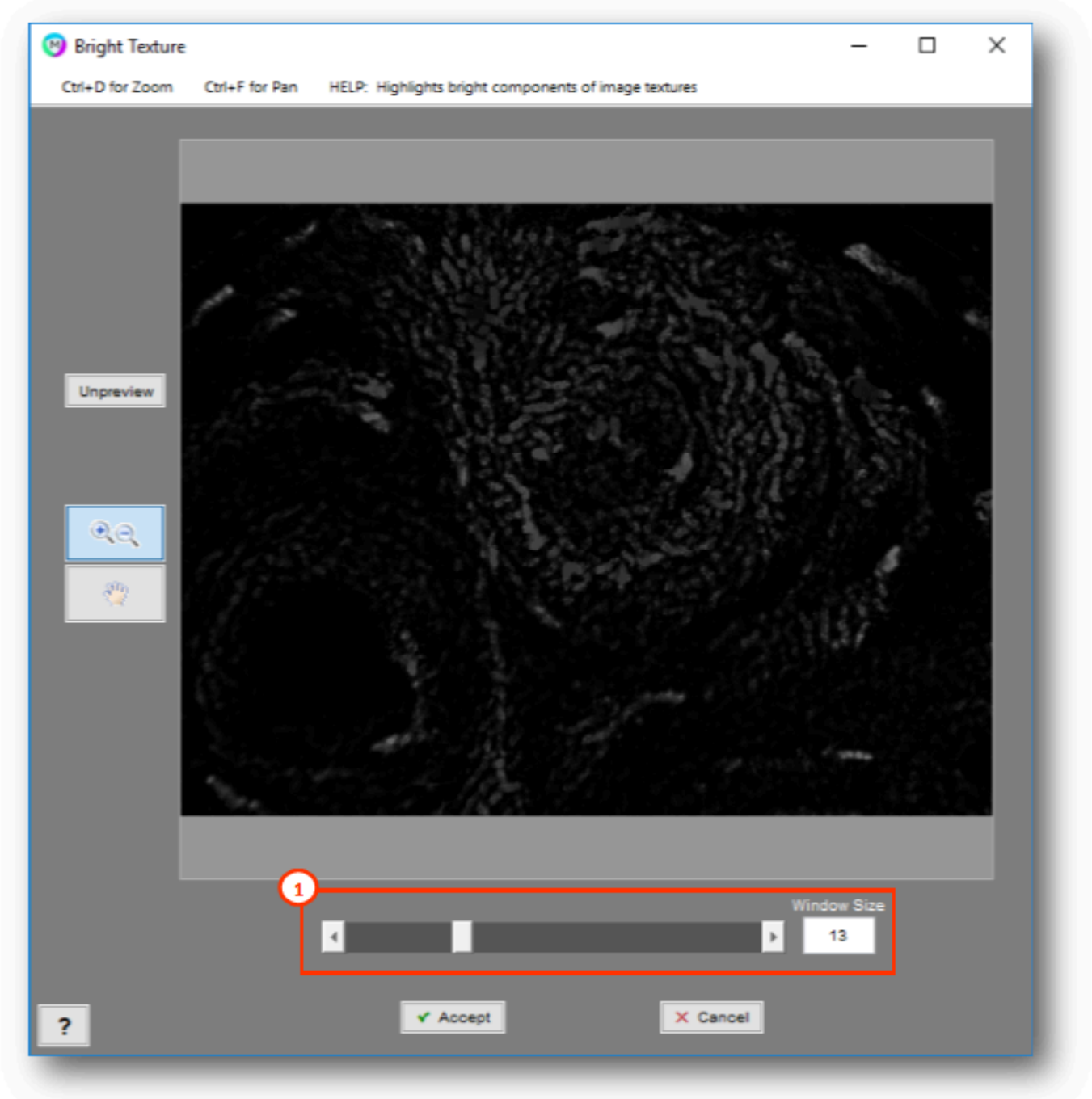
References

[1] Frangi, Alejandro F., Wiro J. Niessen, Koen L. Vincken, and Max A. Viergever. “Multiscale vessel enhancement filtering.” In International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 130-137. Springer Berlin Heidelberg, 1998.

Bright Texture

Pre-Processing > Bright Texture

Highlights bright components of image textures that are no bigger than the specified window size. Also known as a top-hat filter, it computes the grayscale opening (grayscale dilation followed by grayscale erosion) of the Current Image by the specified window size, then subtracts the result from the Current Image.



1. Window Size

Window size (in pixels) to consider about each pixel to calculate the filter. Controls the maximum size of bright feature components to highlight.

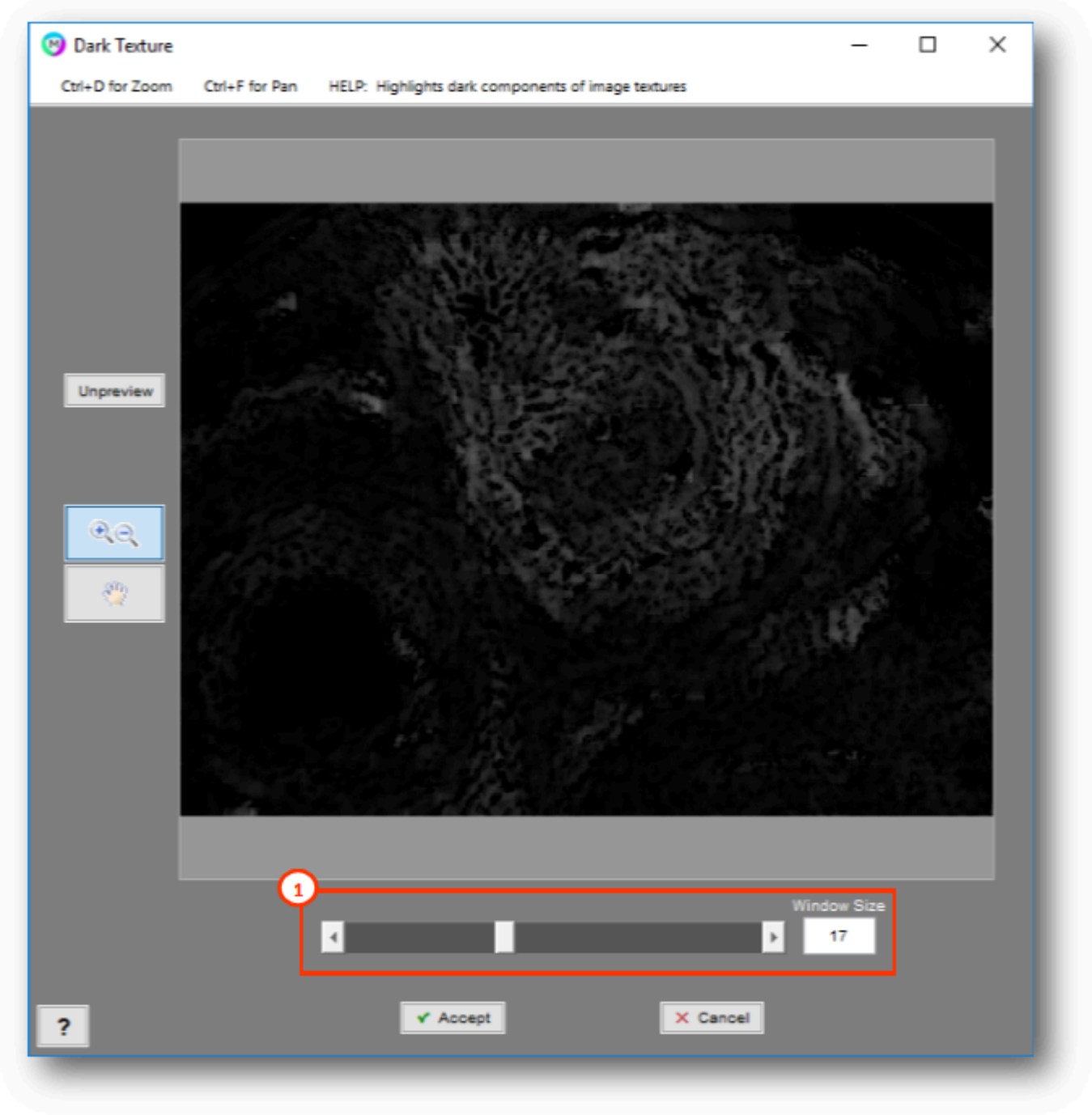
Tips

- Useful for increasing the contrast of flat grey scale features, for example, cells imaged under phase contrast.
- Useful for removing large overexposed regions in fluorescence imaging.
- Starting Window Size: depends on image resolution, start in the middle of the slider, increase to increase brightness, decrease to decrease brightness.

Dark Texture

Pre-Processing > Dark Texture

Highlights dark components of image textures that are no bigger than the specified window size. Also known as a bottom-hat filter, it computes the grayscale closing (grayscale erosion followed by grayscale dilation) of the Current Image by the specified window size, then subtracts the result from the Current Image.



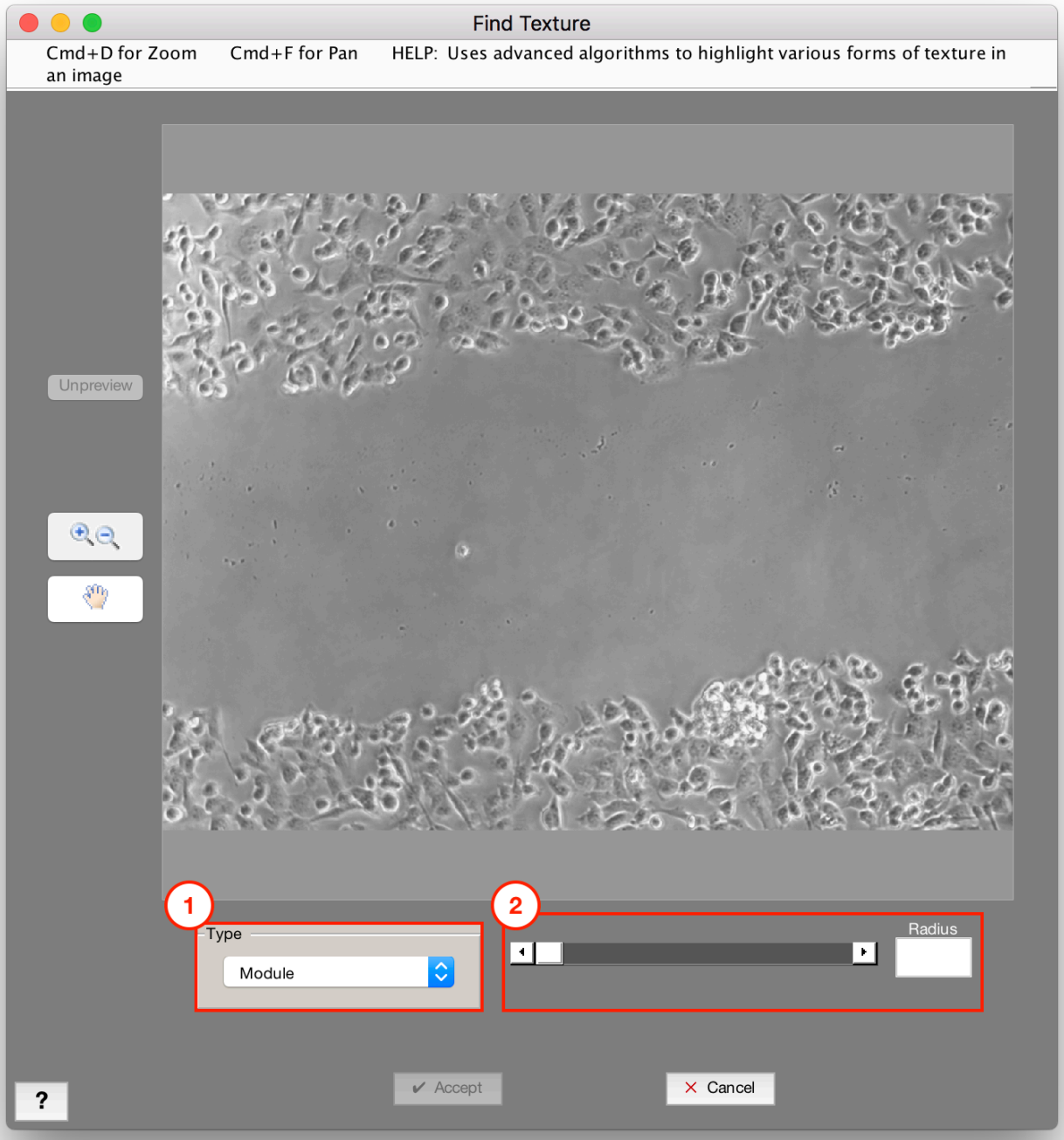
1. Window Size

Window size (in pixels) to consider about each pixel to calculate the filter. Controls the maximum size of dark feature components to highlight.

Advanced Texture

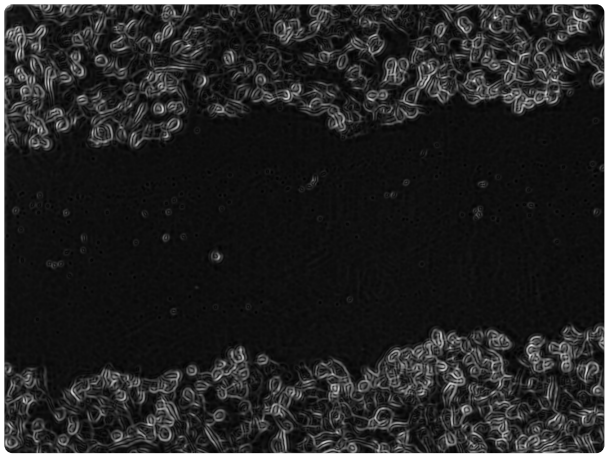
Pre-Processing > Adjust Texture

Uses advanced algorithms to highlight various forms of texture in an image. Options include measuring aspects of each pixel's Hessian matrix [1] and Gabor filters [2].

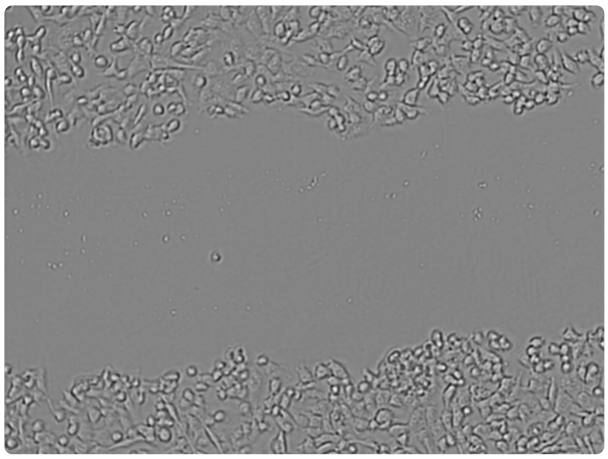


1. Type

- **Module:**

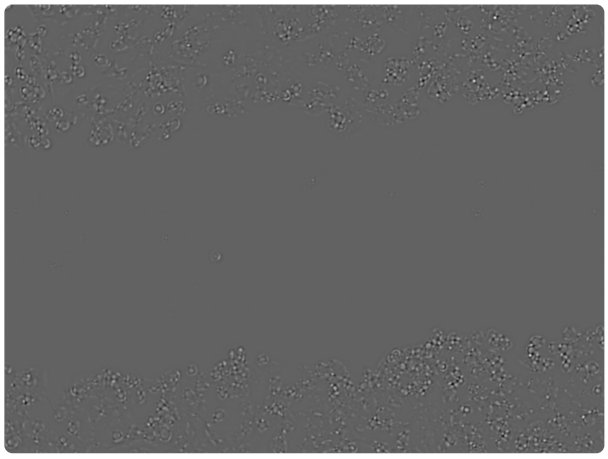


- **Trace:**



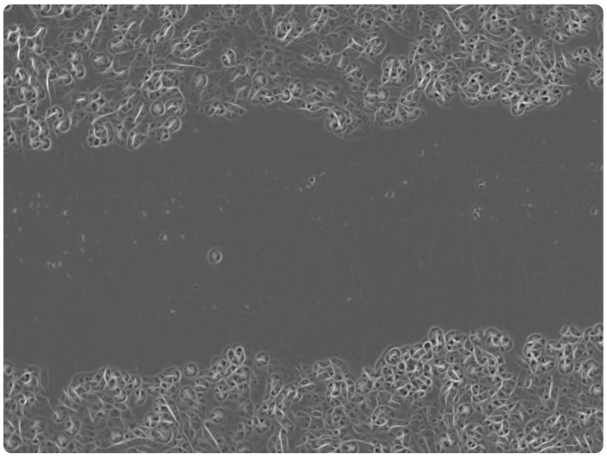
Note that the trace of the hessian matrix is the same as the laplacian operator.

- **Determinant:**

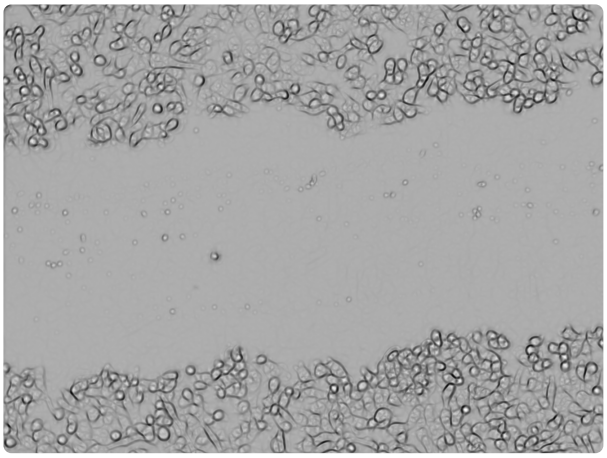


This filter is well-suited to finding corners.

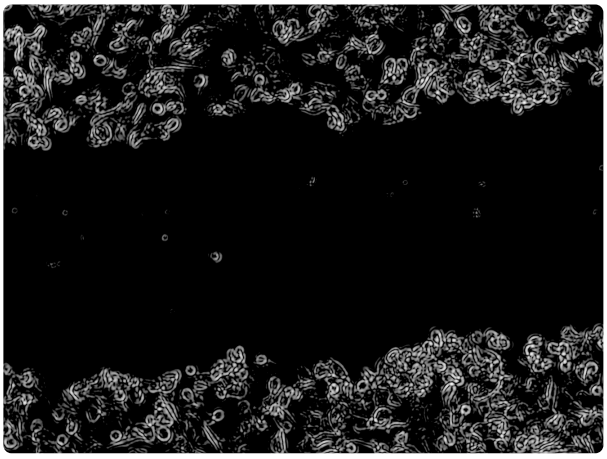
- **First Eigen:**



- **Second Eigen:**

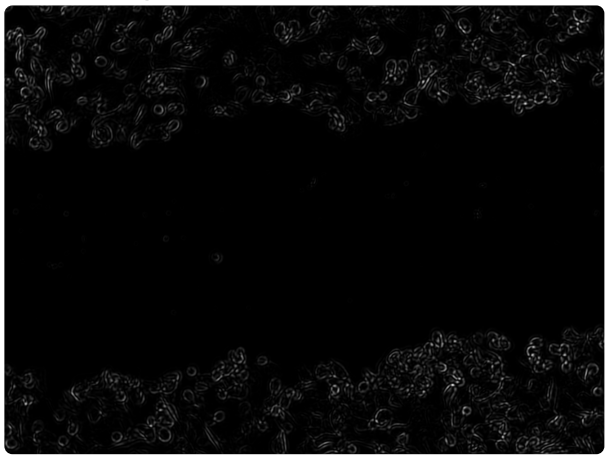


- **Orientation:**

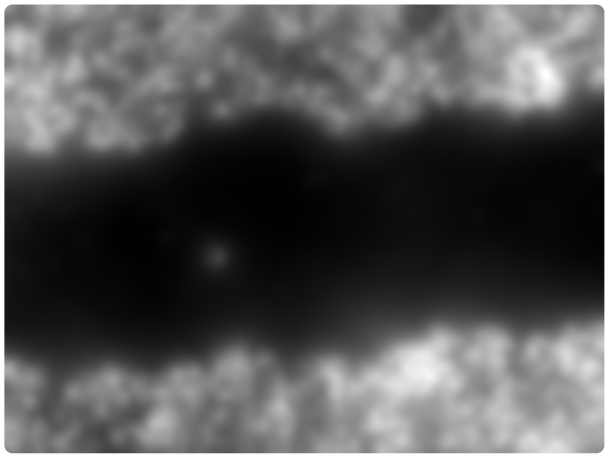


- **Eigen Difference:**

- **Square Eigen Difference:**



- **Gabor:**



2. Radius

Radius (in pixels) to consider about each pixel to calculate the specified filter.

References

[1] D. Eberly, “Ridges in Image and Data Analysis”, Kluwer Academic Publishers, 1996.

[2] Fogel, I.; Sagi, D. (1989). “Gabor filters as texture discriminator”. Biological Cybernetics. 61 (2).

Similarity

Contains filters that detect similarity and symmetry within an image or between two images.

- Pattern Mapping
- Orthogonal Correlate
- Symmetry Mapping
- *Cross-Correlate

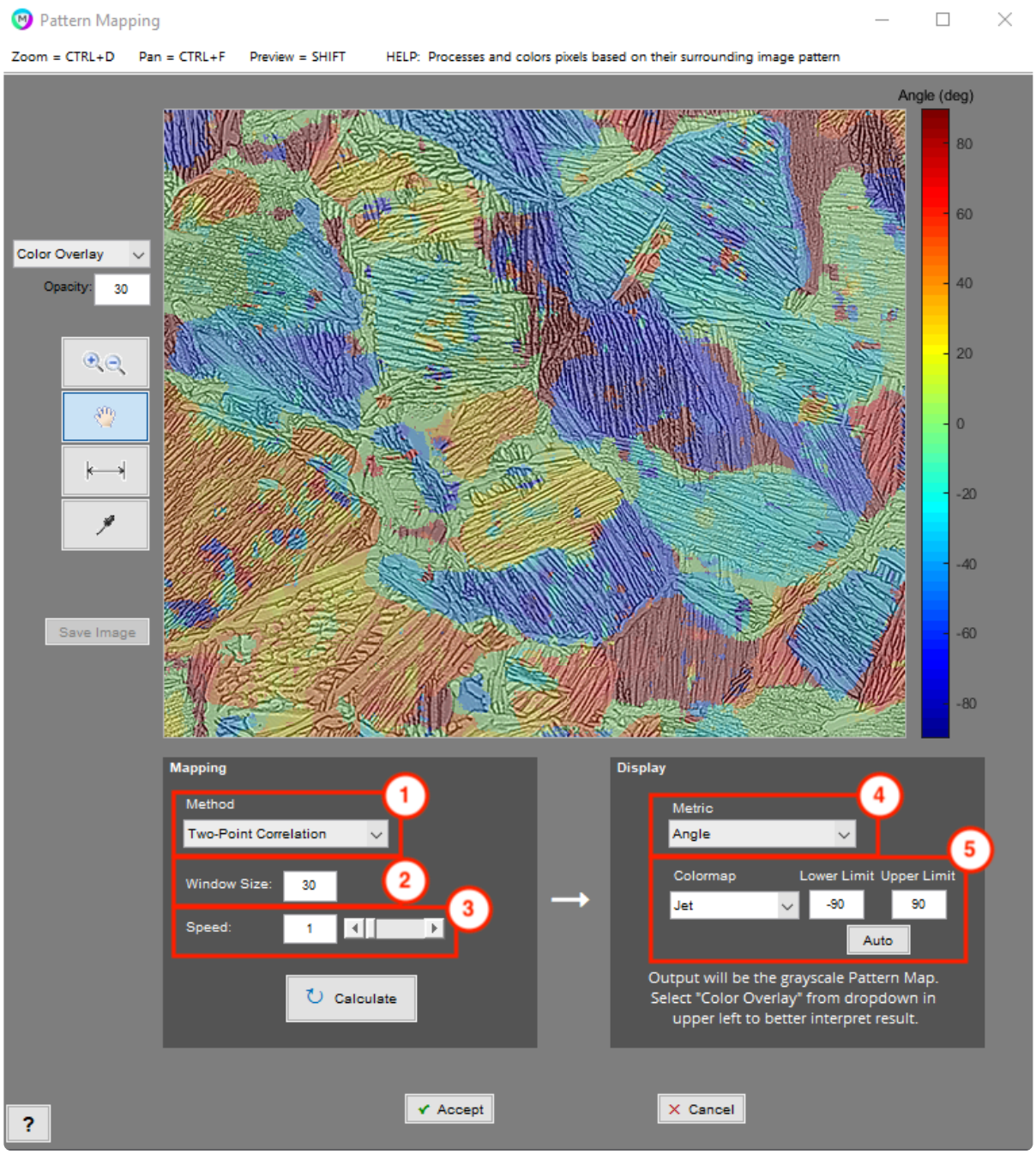
Pattern Mapping

Pre-Processing > Similarity > Pattern Mapping

Processes and colors pixels based on their surrounding image pattern. Produces a grayscale map of the pattern indicator for further processing into an ultimate selection.

Available tools for pattern analysis are local two-point correlation properties, mapping of selected FFT intensities, and matching of local FFTs to selected references. This function can take some time to complete depending on the size of the image and Window Size parameter.

After first clicking “Calculate”, a parallel-pool of CPU resources will be created which may take several seconds. This parallel-pool is good for 30 minutes and will not have to be re-created within that time. It is recommended that this function not be run on images larger than 1200×900, and that the window size be kept to 40 pixels or less.



1. Method

- **Two-Point Correlation:** Calculates the two-point correlation function at each pixel, using a surround window of a specified size. The orientation, size, and shape of the “hotspot” of a pixel-window’s two-point correlation tell you information about the window’s local directionality, the extent of that directionality, and the size of features within the window.

- **FFT Mapping:** Allows you to define a mask to place around the FFT taken about each pixel, and either sum the intensities or take the peak intensity within the mask. This measurement is then assigned to each pixel. In essence, you are mapping out traits about each pixel’s local FFT, which can reveal local pattern differences in the image. The size of the window taken about the click to define the mask is determined by Window Size. It is often advised to set a large Window Size prior to

clicking to define the mask, and then reducing the Window Size to something much smaller to perform the mapping.

- **FFT Matching:** Allows to define a fixed number of reference FFTs (i.e., “fingerprints”) by clicking points within the image. The size of the window taken about each click to define the “fingerprints” is determined by Window Size . Each pixel’s local FFT is then compared to each “fingerprint”, and labeled according to which “fingerprint” it matched the best.

2. Window Size

Window size to consider about each pixel for the pattern map

3. Speed

Controls the speed of the pattern mapping by increasing or decreasing the degree of downsampling

4. Metric

Mapped parameter to display

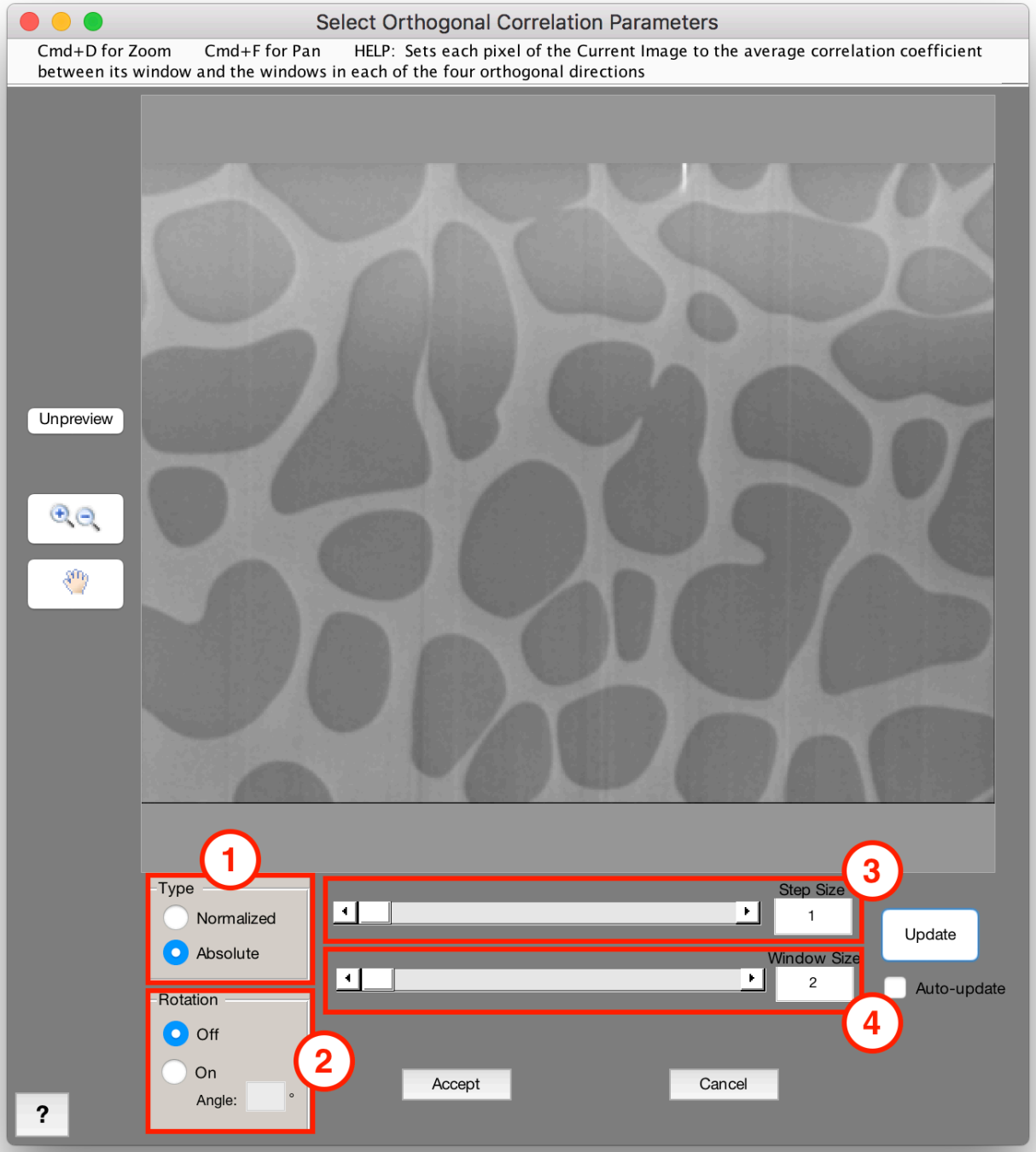
5. Colormap and Limits

Colormap to use for display, along with upper and lower limits for the colormap. Click “Auto” to automatically set the default limits.

Orthogonal Correlate

Pre-Processing > Matching > Orthogonal Correlate

Sets each pixel of the Current Image to the average correlation coefficient between its window and the windows in each of the four orthogonal directions. Can be used to identify regions of certain symmetries in “Normalized” mode, and as a contrast enhancement technique in “Absolute” mode.



1. Type

Sets type for orthogonal correlation finding.

- **Normalized:** Calculate normalized correlation coefficient at each pixel. Helpful for highlighting out regions of local similarity.

- **Absolute:** Calculate absolute correlation coefficient at each pixel. Helpful for enhancing image contrast.

2. Rotation

Choose whether rotate image about its center before correlating each pixel with its four orthogonal windows.

- **Off:** Do not rotate image before choosing orthogonal windows about each pixel
- **On:** Rotate image before choosing orthogonal windows about each pixel. Set the angle for rotation.

3. Step Size

Step size to move in either direction to choose orthogonal windows. For contrast enhancement, larger step size can result in more blurring, and should be kept as close to 1 as possible.

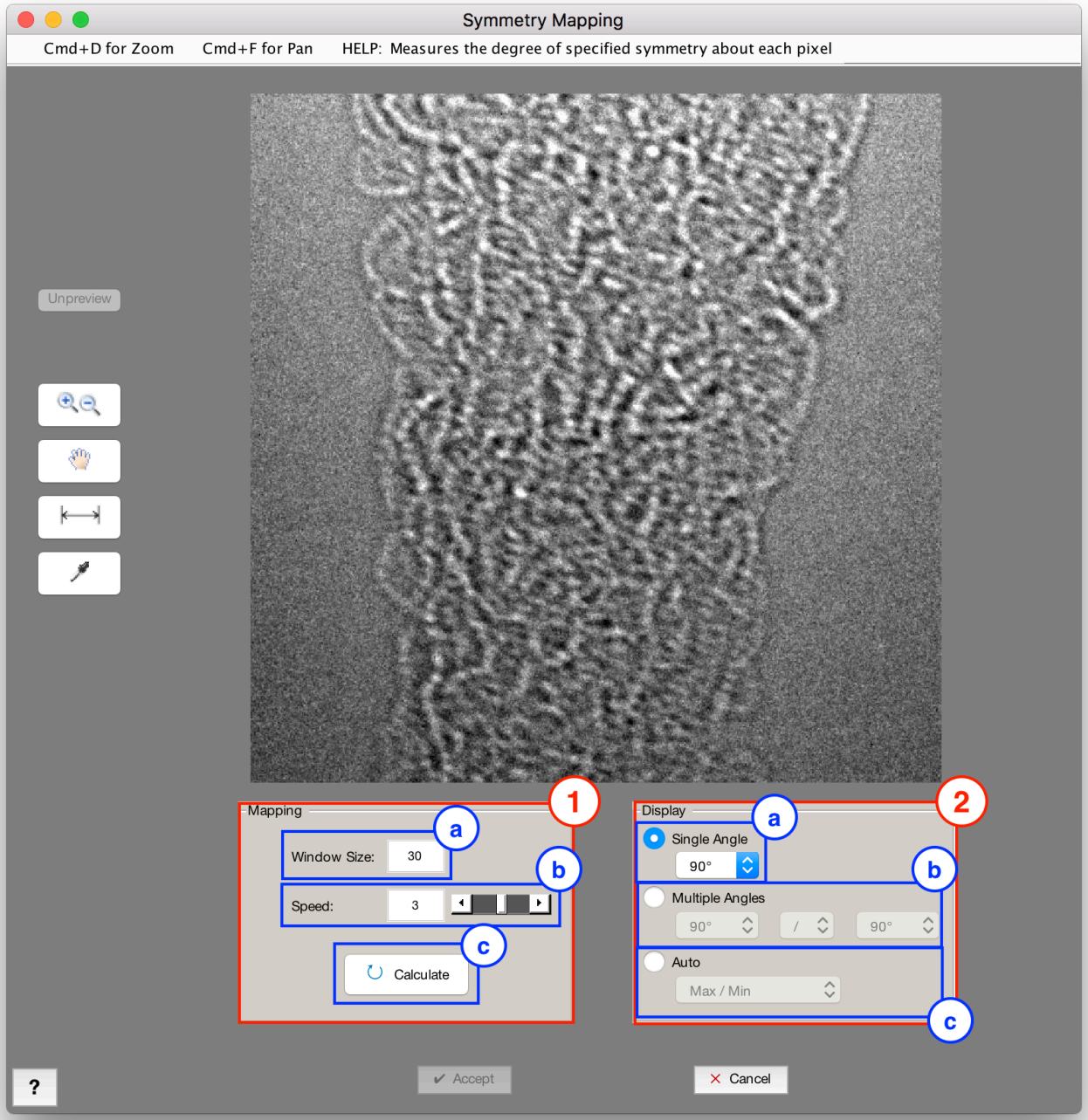
4. Window Size

Size of the orthogonal windows. For contrast enhancement, larger window size can result in more blurring, and should be kept as close to 2 as possible.

Symmetry Mapping

Pre-Processing > Matching > Symmetry Mapping

Measures the degree of specified symmetry about each pixel. Bright pixels in the symmetry-filtered image indicate pixels which had high similarity of the specified angles.



1. Mapping

a. Window Size

Window size to consider about each pixel for the pattern map

b. Calculate

Calculate symmetry map

2. Display

a. Single Angle

Display symmetry map for single angle. Options are 90°, 180°, and 270°. For example, selecting 90° will set each pixel based on the amount of 90° rotational symmetry about the pixel.

b. Multiple Angles

Display symmetry map for arithmetic between multiple angles. Set arithmetic for symmetry map from dropdowns.

c. Auto

Auto-display symmetry map. Select a method for auto-display.

*Cross-Correlate

Pre-Processing > Matching > *Cross Correlate

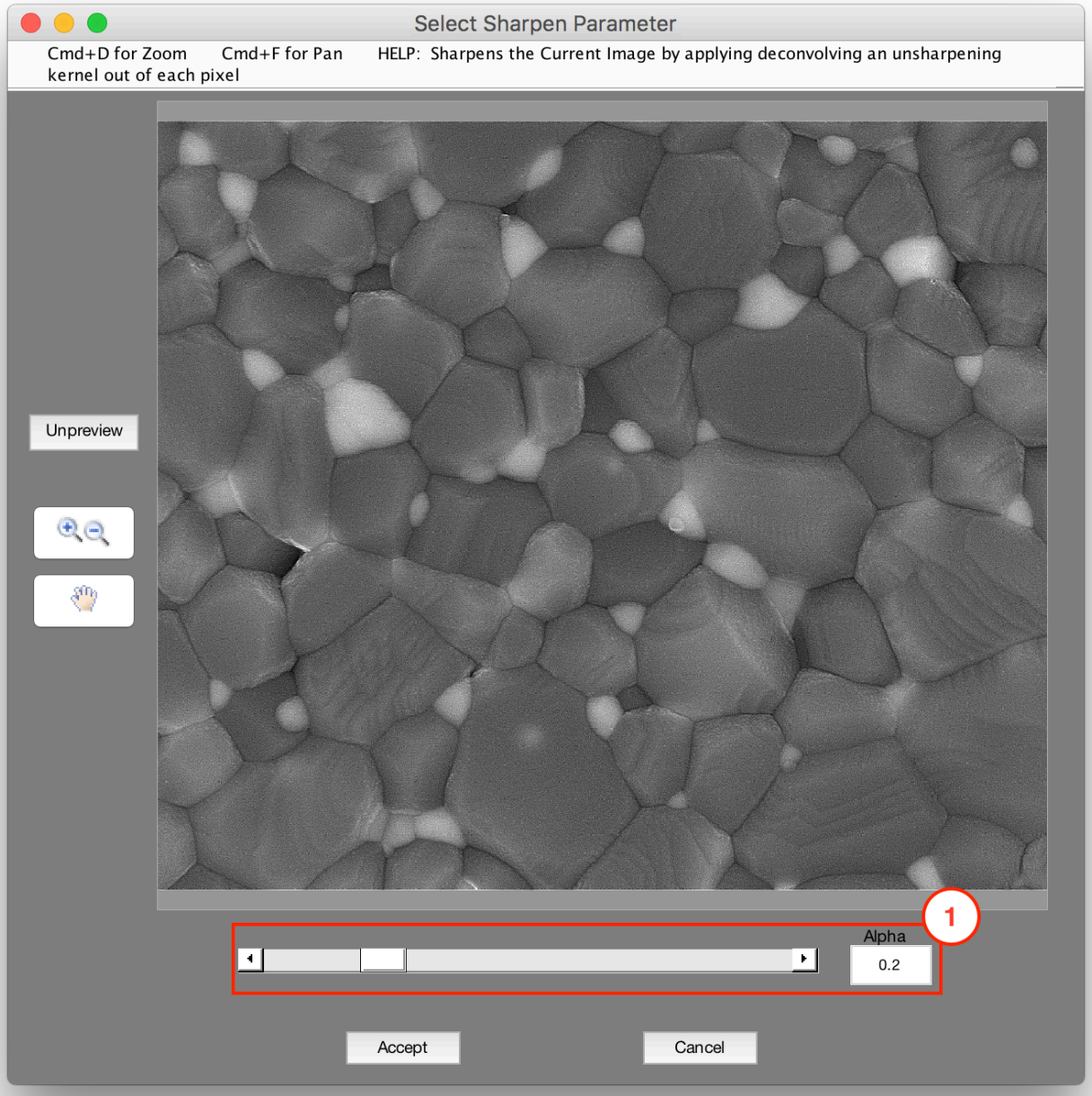
Requires B/W Companion Image

Correlates each pixel in the Current Image with the Companion Image.

Sharpen

Pre-Processing > Sharpen

Sharpens the Current Image by deconvolving an unsharpening kernel out of each pixel.



1. Alpha

Parameter which controls sharpness. Higher values produce more sharpening.

Tips

Useful for restoring the contrast of fine features dulled by noise removing filters.

Frequency

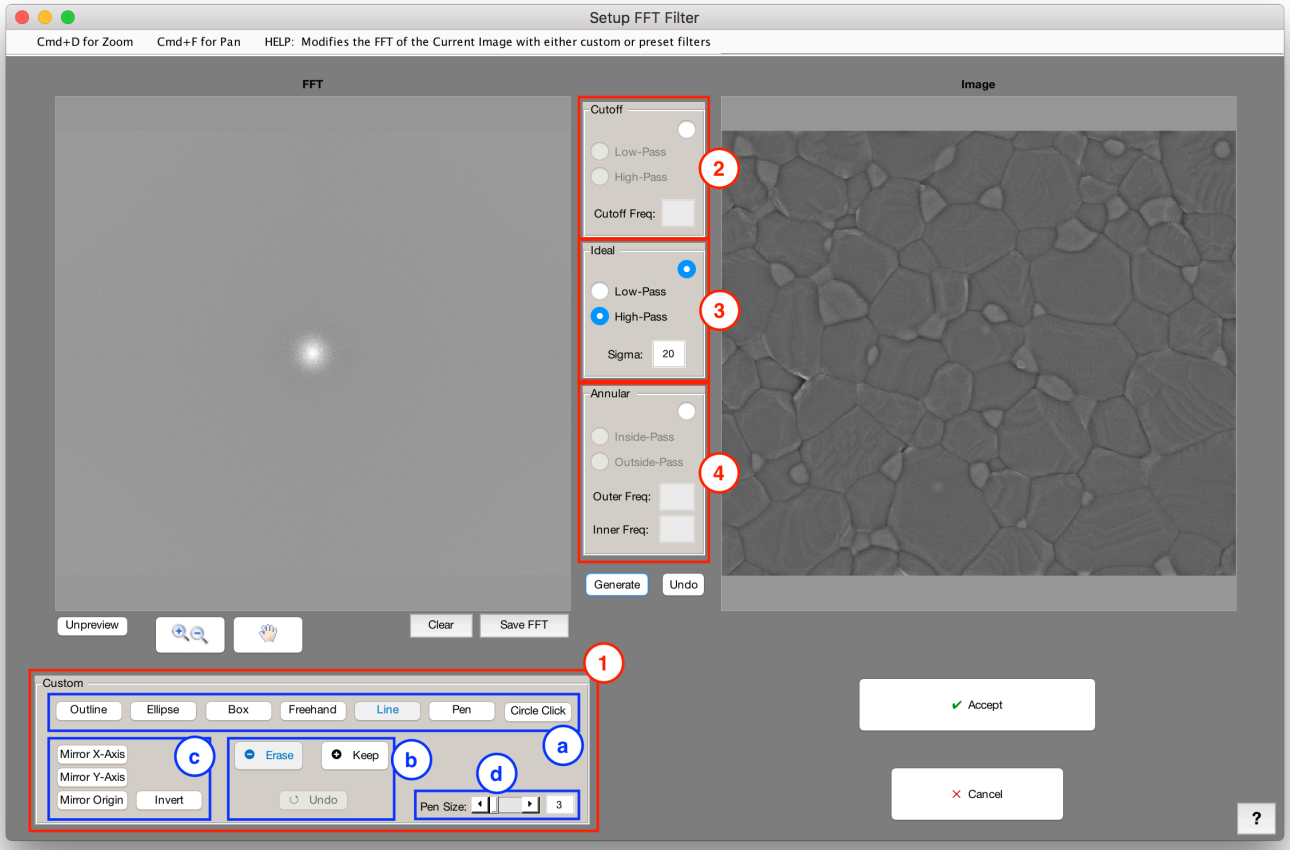
Contains filters that work by manipulating the image's FFT (fast-Fourier transform), which expresses pixels in frequency space.

- FFT Filter
- Make FFT
- Apply Stored FFT Filter

FFT Filter

Pre-Processing > Frequency > FFT Filter

Modifies the Fourier transform of the Current Image with either custom or preset filters. Has many applications, ranging from background removal to edge-finding to artifact reduction.



1. Custom

a. Selection Tools

- **Outline:** Fill/erase a polygon area by clicking to create nodes. Click the last node to close the polygon.
- **Ellipse:** Click, drag, and release to fill/erase an elliptical/circular selection
- **Box:** Click, drag, and release to fill/erase a rectangular/square selection
- **Freehand:** Click, drag, and release to fill/erase a freehand selection. Selection closes automatically.

- **Line:** Click, drag, and release to fill/erase a straight line selection
- **Pen:** Click, drag, and release to fill/erase a pen (freehand line) selection
- **Circle Click:** Click a point to define a center of circular selection. Pixel diameter of circles is determine by ***Pen Size***.

b. Process Selection

- **Erase:** Erase the selection from the image's FFT
- **Keep:** Keep the selection in the image's FFT (discards the rest if no other selection has been processed, adds the selected intensity back if one has been processed)
- **Undo** Undo the last fill/erase operation

b. Mirror/Invert Selection

- **Mirror X-Axis:** Mirror current filter over X-axis
- **Mirror Y-Axis:** Mirror current filter over Y-axis
- **Mirror Origin:** Mirror current filter over origin
- **Invert:** Invert current filter

b. Pen Size

Controls the thickness (in pixels) of the Line and Pen selection tools, as well as the diameter (in pixels) of the circles created by the Circle Click tool

2. Cutoff

Use a cutoff filter. Places a hard-edged circle filter around the center of the FFT.

- **Low-Pass:** Keep frequencies within the filter
- **High-Pass:** Keep frequencies outside filter
- **Cutoff Frequency:** Size of the filter circle in frequency space

3. Ideal

Use an ideal filter. Places a 2D Gaussian filter around the center of the FFT.

- **Low-Pass:** Keep frequencies within the filter

- **High-Pass:** Keep frequencies outside filter
- **Sigma:** Proportional to the size of the filter circle in frequency space

4. Annular

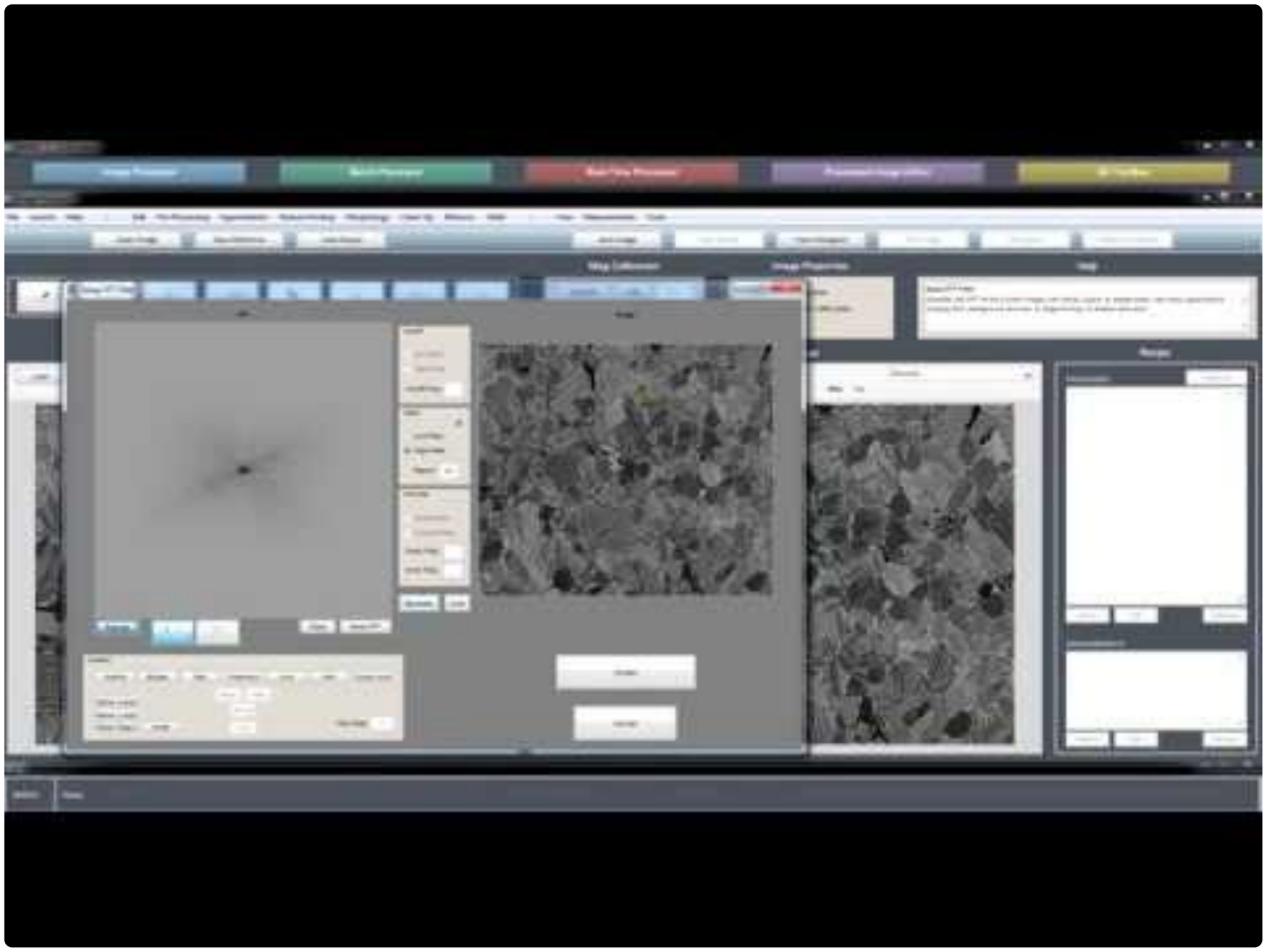
Use an annular filter. Places a ring filter around the center of the FFT.

- **Inside-Pass:** Keep frequencies within the filter
- **Outside-Pass:** Keep frequencies outside the filter
- **Outer Frequency:** Size of the outer ring diameter
- **Inner Frequency:** Size of the inner ring diameter

Tips

- The points on the FFT will always be perpendicular to the direction of the artifact you wish to remove.
- Example: SEM curtaining is causing vertical artifacts in the image, remove horizontal points in FFT to remove curtaining. (See Tutorial Video)

Tutorial



https://www.youtube.com/embed/X2GzqYL_i88?rel=0

https://www.youtube.com/embed/X2GzqYL_i88?rel=0

Make FFT

Pre-Processing > Frequency > Make FFT

Generates FFT of the Current Image. Useful for building custom filters with subsequent recipe steps, and then applying the filter with “Apply Stored FFT Filter” after setting filter as the Companion Image.

Apply Stored FFT Filter

Pre-Processing > Frequency > Apply Stored FFT Filter

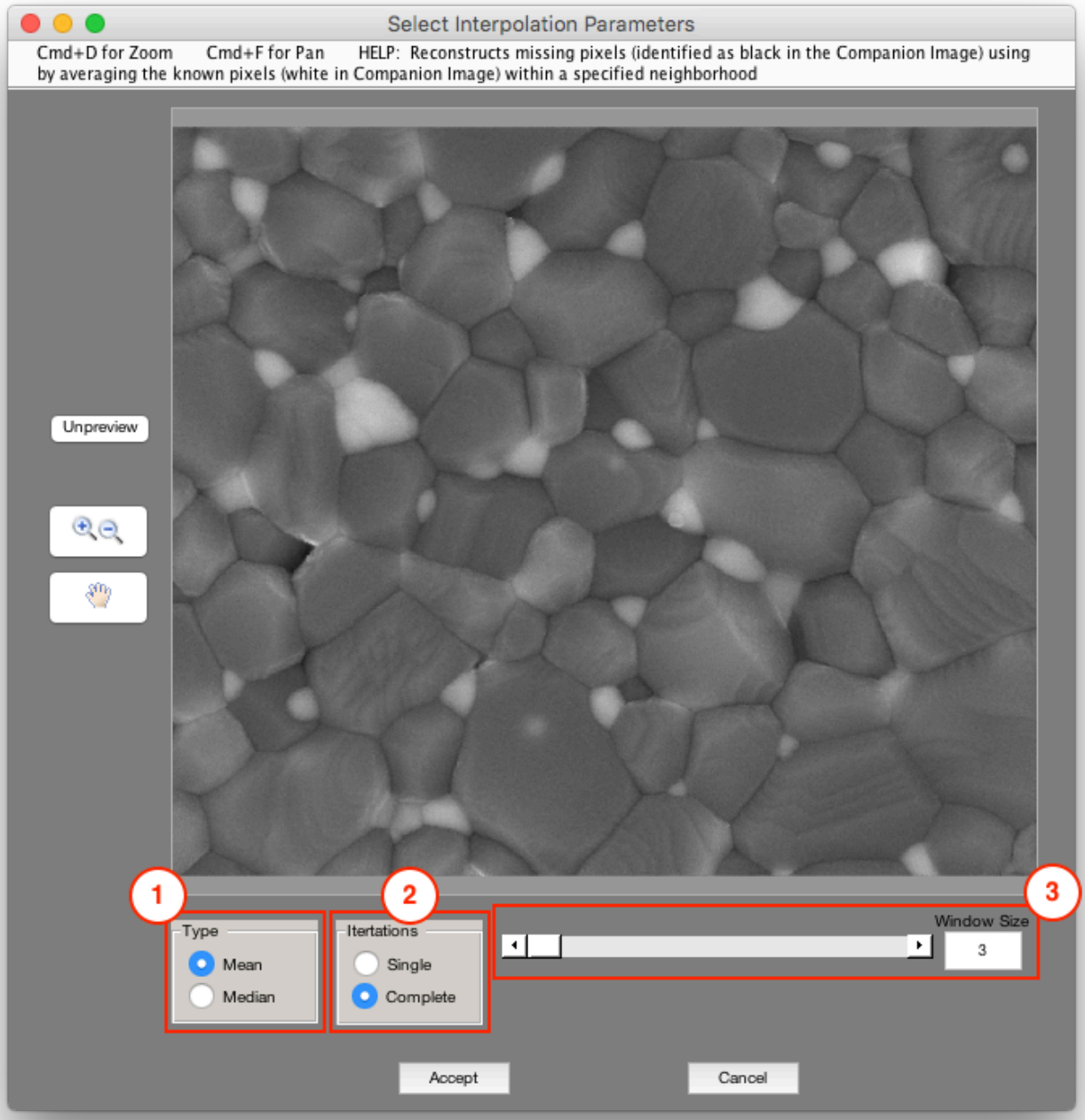
Applies the Companion Image as an FFT filter to the Current Image. Dark or black pixels in the filter will remove those portions of the image's FFT. Current Image should be the image to be filtered, and Companion Image should hold the filter to apply in FFT space.

*Grayscale Interpolation

Pre-Processing > *Grayscale Interpolation

Requires B/W Companion Image

Reconstructs missing pixels (identified as selection in the Companion Image) using by averaging the known pixels (empty in Companion Image) within a specified neighborhood.



1. Type

- **Mean:** Uses mean of known pixels within window to replace unknown pixels
- **Median:** Uses median of known pixels within window to replace unknown pixels

1. Iterations

- **Single:** Interpolates missing pixels once though the image
- **Complete:** Interpolates missing pixels until there are none left in the image

3. Window Size

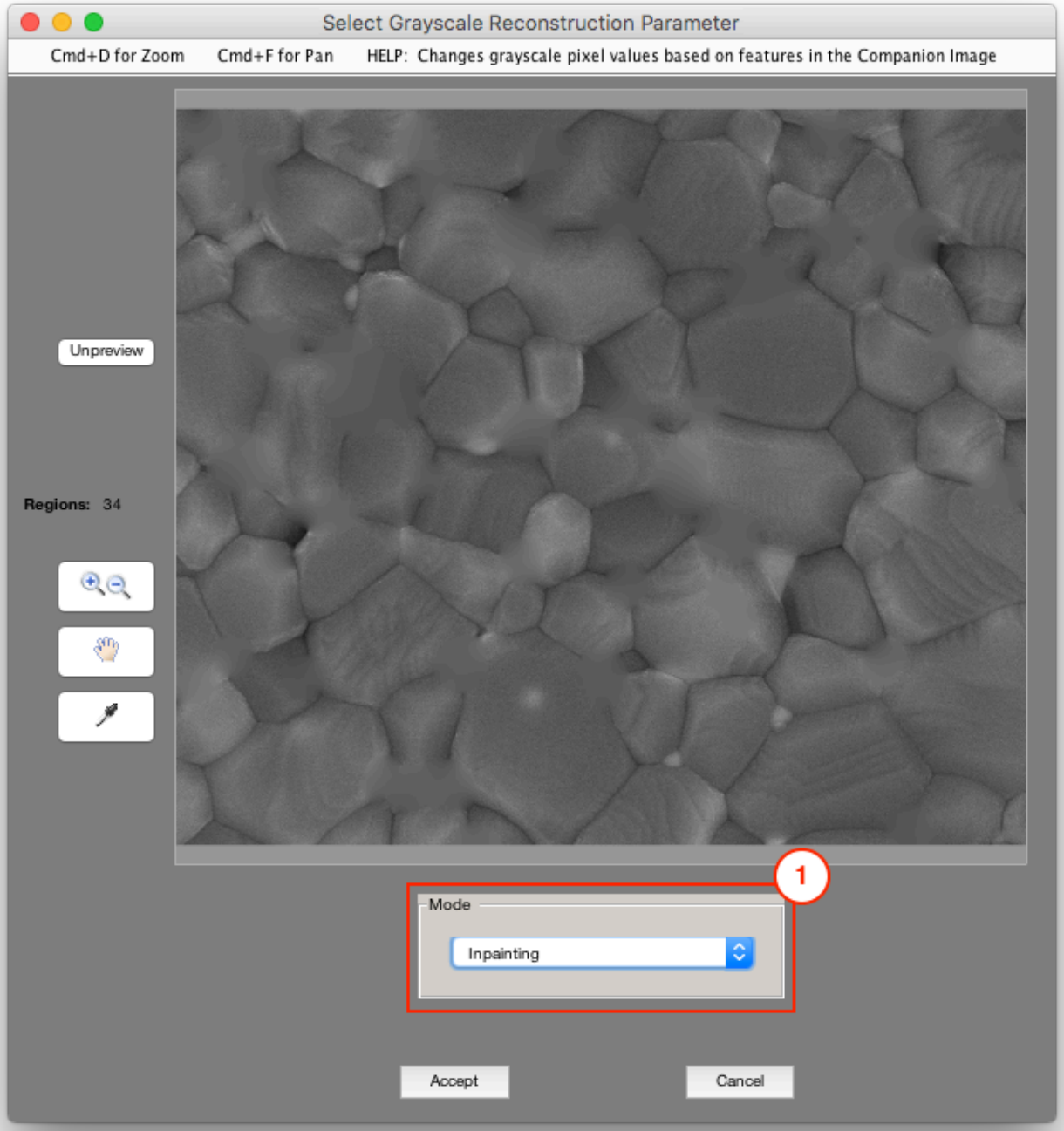
Window size to consider about each pixel for the mean or median calculation

*Grayscale Reconstruction

Pre-Processing > *Grayscale Reconstruction

Requires B/W Companion Image

Changes grayscale pixel values based on features in the Companion Image. Available modes are morphological reconstruction, impose minima (ensures that minima in the grayscale image only occur at feature pixels), or set pixels equal to the mean, median, or standard deviation of the pixels within each feature.



1. Mode

- **Inpainting:** Smoothly interpolates inward within each marked region from the pixel values on the outer boundary of the region. Works by computing the discrete Laplacian over the regions and solves the Dirichlet boundary value problem for each [1].

- **Morphological Reconstruction:** Performs morphological reconstruction using the defined markers [2]

- **Impose Minima:** Modifies the image so that it only has local minima wherever markers are defined
- **Fill With Mean:** Fills the pixels within the marker regions with the mean pixel value of that region
- **Fill With Median:** Fills the pixels within the marker regions with the median pixel value of that region
- **Fill With Mode:** Fills the pixels within the marker regions with the mode pixel value of that region
- **Fill With StdDev:** Fills the pixels within the marker regions with the standard deviation pixel value of that region
- **Fill With Min:** Fills the pixels within the marker regions with the minimum pixel value of that region
- **Fill With Max:** Fills the pixels within the marker regions with the maximum pixel value of that region

References

[1] Pérez, Patrick, Michel Gangnet, and Andrew Blake. “Poisson image editing.” In ACM Transactions on Graphics (TOG), vol. 22, no. 3, pp. 313-318. ACM, 2003. Harvard

[2] Vincent, L., “Morphological Grayscale Reconstruction in Image Analysis: Applications and Efficient Algorithms,” IEEE Transactions on Image Processing, Vol. 2, No. 2, April, 1993, pp. 176-201.

Deep Learning

Contains functions for applying AI-based models for feature detection. Models are created in the Deep Learning Trainer.

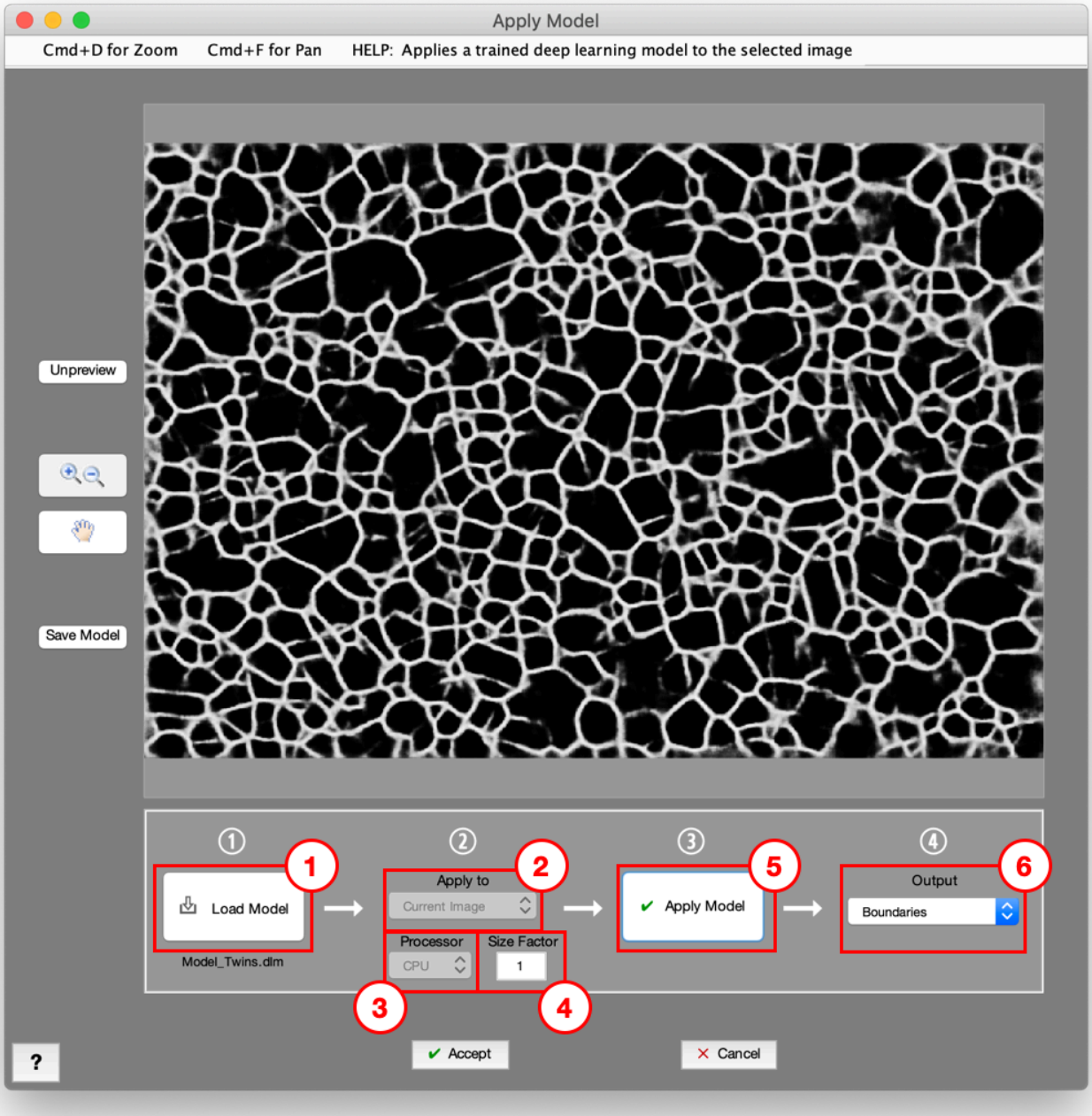
Functions

- Apply Model
- Call Output

Apply Model

Deep Learning > Apply Model

Applies a trained deep learning model to the selected image. Output can be a probability map of any of the trained layers, or a layer map. A probability map indicates each pixel's likelihood to belong to that layer. A layer map represents the most likely layer for each pixel.



1. Load Model

Load deep learning model. These are created and saved from the Deep Learning Trainer.

2. Apply To

Choose image to apply model to.

- **Color Image:** Model is applied to the originally opened color image (only available for color images).
- **Current Image:** Model is applied to Current Image.

3. Processor

Processor to apply model with.

- **GPU:** Apply model on the GPU (strongly recommended — see Deep Learning System Requirements for more information).
- **CPU:** Apply model on the CPU.

4. Size Factor

Resize factor prior to applying model (0-1). Image will be resized according to the this factor, model applied, then results upsampled back to original size. It is recommended to use the same size factor as was used to train the model.

5. Apply Model

Apply deep learning model.

6. Output

Choose image to display and output.

- **Layer Map:** Represents the most likely Layer classification for each pixel.
- **Other Options:** Probability maps of the trained Layers. A probability map indicates each pixel's likelihood to belong to that layer.

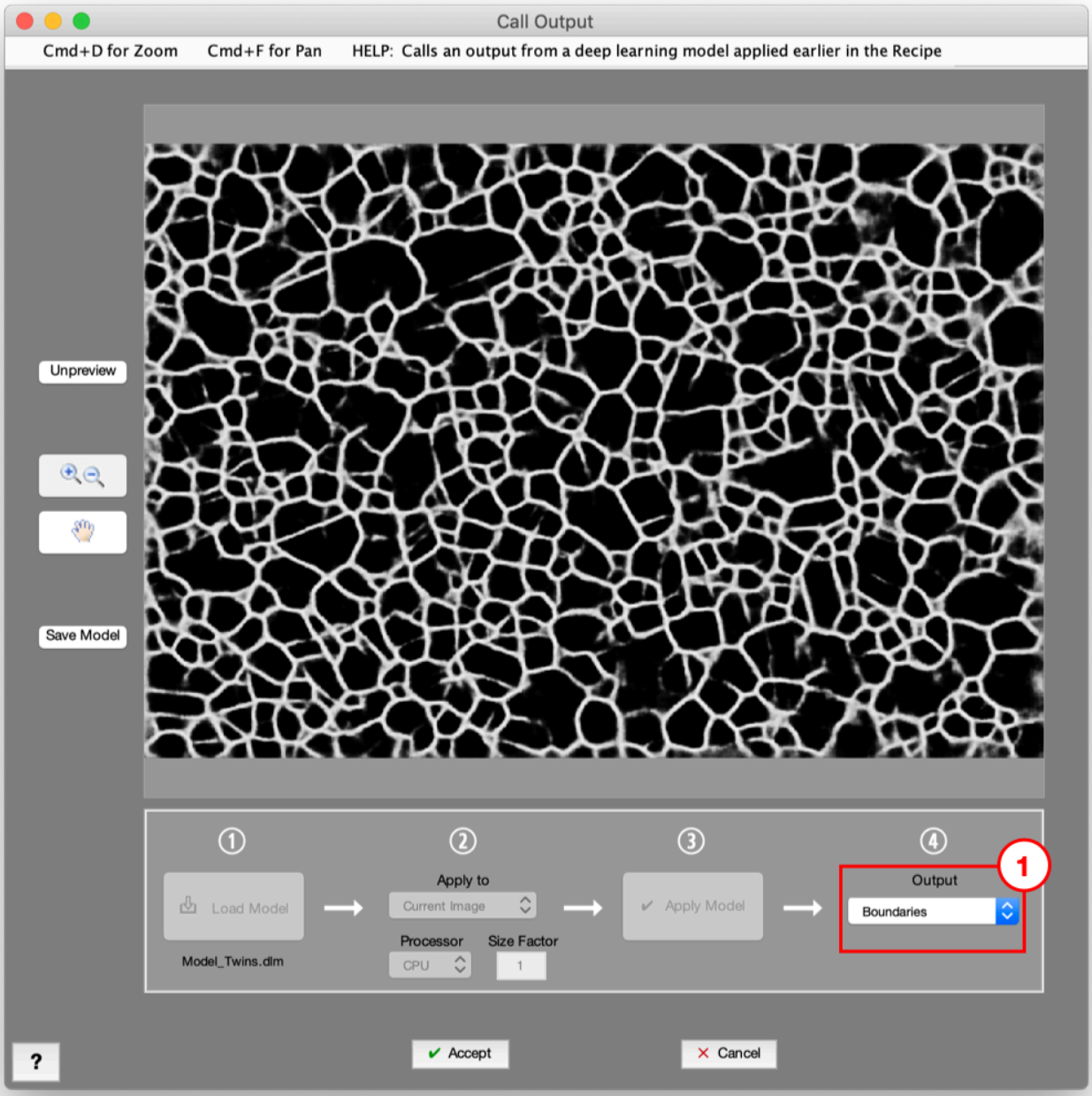
Example

Applying a deep learning model

Call Output

Deep Learning > Call Output

Calls an output from a deep learning model applied earlier in the Recipe. This is useful for performing separate processing workflows on different probability maps from a single Apply Model step.



1. Output

Choose image to display and output

- **Layer Map:** Represents the most likely Layer classification for each pixel
- **Other Options:** Probability maps of the trained Layers. A probability map indicates each pixel's liklihood to belong to that layer.

Segmentation

Contains functions for selecting features in the image. These operate on grayscale images and range from simple such as Basic Threshold, to more advanced such as Adaptive Threshold and Watershed.

Functions

- Invert
- Blank
- Manual Edit

Threshold

- Basic Threshold
- Range Threshold
- Adaptive Threshold
- E-M Threshold
- *Local Threshold

Edges

- Watershed
- Find Edges
- Find Circles
- Find Lines
  - Find Text
  - Find Facial Features

Snap

- Auto Segmentation
- *Region Grow
- *Fast Marching Method
- *Active Contour

Extrema

- Find Global Maximum
- Find Global Minimum
- Find Local Maxima
- Find Local Minima

Invert

Segmentation > Invert

Inverts the Current Image. Grayscale pixels become 255 minus their value. Selected pixels become empty, and empty pixels become selected.

Tips

When segmenting particles from matrix in a composite material, often times it is easier to accurately select the sample matrix and invert to result in particle selection.

Blank

Segmentation > Blank

Creates a blank segmentation. Useful for manually creating seeds with “Manual Edit” for use with steps such as “Region Grow”, “Active Contour”, and “Local Threshold

Tips

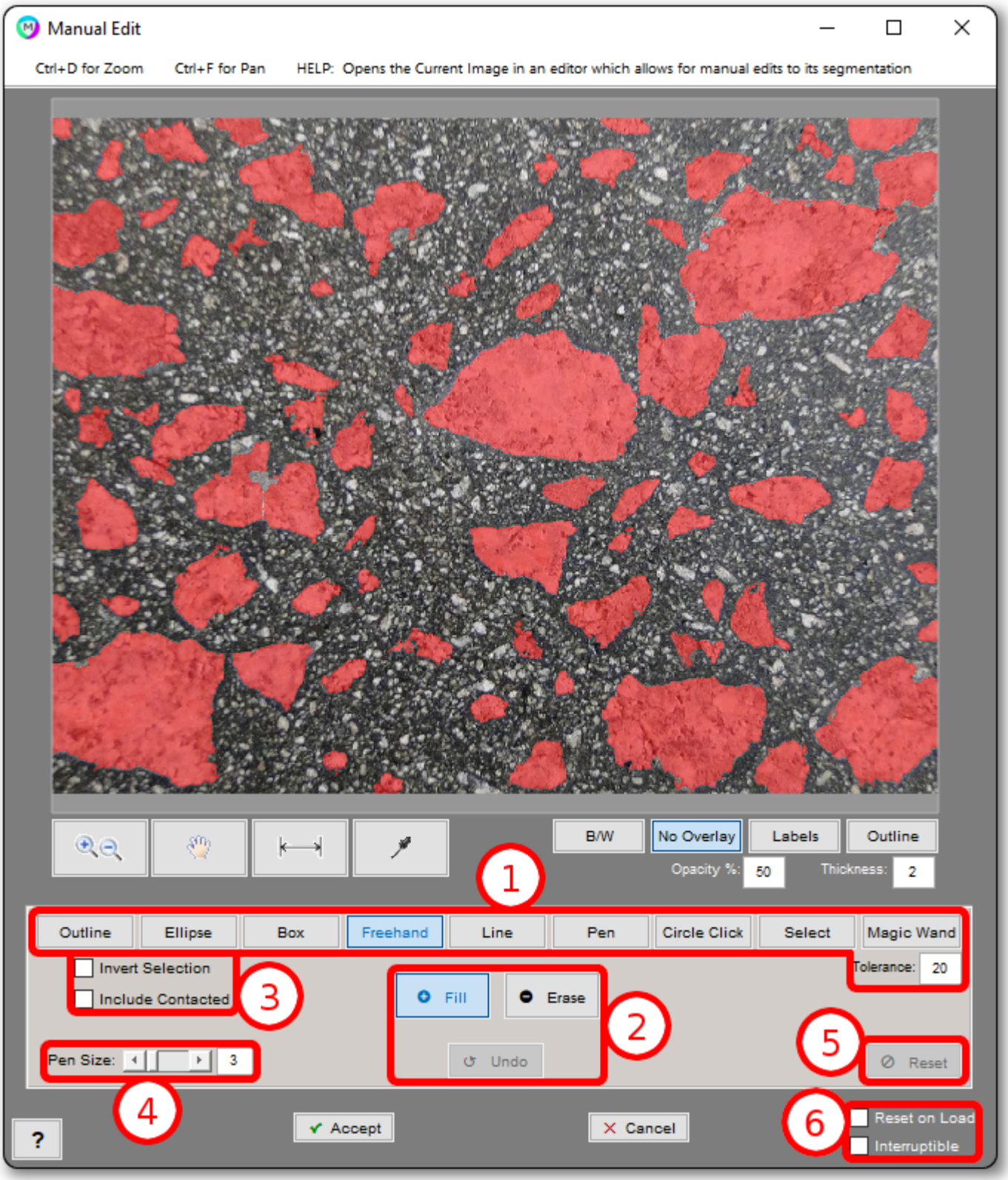
- Use in combination with Manual Edit tools and Edit>Position tools to create masks used for ‘Math’ steps.
- Example: Blank>Translate: can generate a 1 pixel boundary line that you can use to isolate features touching a single edge.

Manual Edit

Segmentation > Manual Edit

Opens the Current Image in an editor which allows for manual edits to its segmentation. Here you can add missed features and remove artifacts. Also useful for manually defining a region of interest (ROI), which when set in a memory image, can be used to restrict your feature selection to a certain area in the image.

This function requires the previous step to produce a segmentation, not a grayscale image. Accordingly, it also cannot be the first step in a recipe; to achieve a similar effect, precede it with a Segmentation -> Blank step.



1. Selection Tools

- **Outline:** Fill/erase a polygon area by clicking to create nodes. Click the last node to close the polygon.

- **Ellipse:** Click, drag, and release to fill/erase an elliptical/circular selection
- **Box:** Click, drag, and release to fill/erase a rectangular/square selection
- **Freehand:** Click, drag, and release to fill/erase a freehand selection. Selection closes automatically.
- **Line:** Click, drag, and release to fill/erase a straight line selection
- **Pen:** Click, drag, and release to fill/erase a pen (freehand line) selection
- **Circle Click:** Click a point to define a center of circular selection. Pixel diameter of circles is determine by ***Pen Size***.
- **Select:** Click a feature to fill/erase it.
- **Magic Wand:** Click a point to magic-wand select a region around it based on the surrounding grayscale pixel values. Tolerance defines how different neighboring pixel intensities can be included in the region. Higher tolerance leads to larger regions.

See Keyboard Shortcuts for shortcuts relevant to manual editing tools.

2. Tool Mode

- **Fill:** Fill the selection (make it black in B/W view)
- **Erase:** Erase the selection (make it white in B/W view)
- **Undo** Undo the last fill/erase operation

3. Options

- **Invert Selection:** Invert the current selection tool operation (select/erase everything else)
- **Include Contacted:** Include entire features that even partially contact the current selection

4. Pen Size

Controls the thickness (in pixels) of the Line and Pen selection tools, as well as the diameter (in pixels) of the circles created by the Circle Click tool

5. Reset

Clears all edits, both fills and erasures, made to the selection.

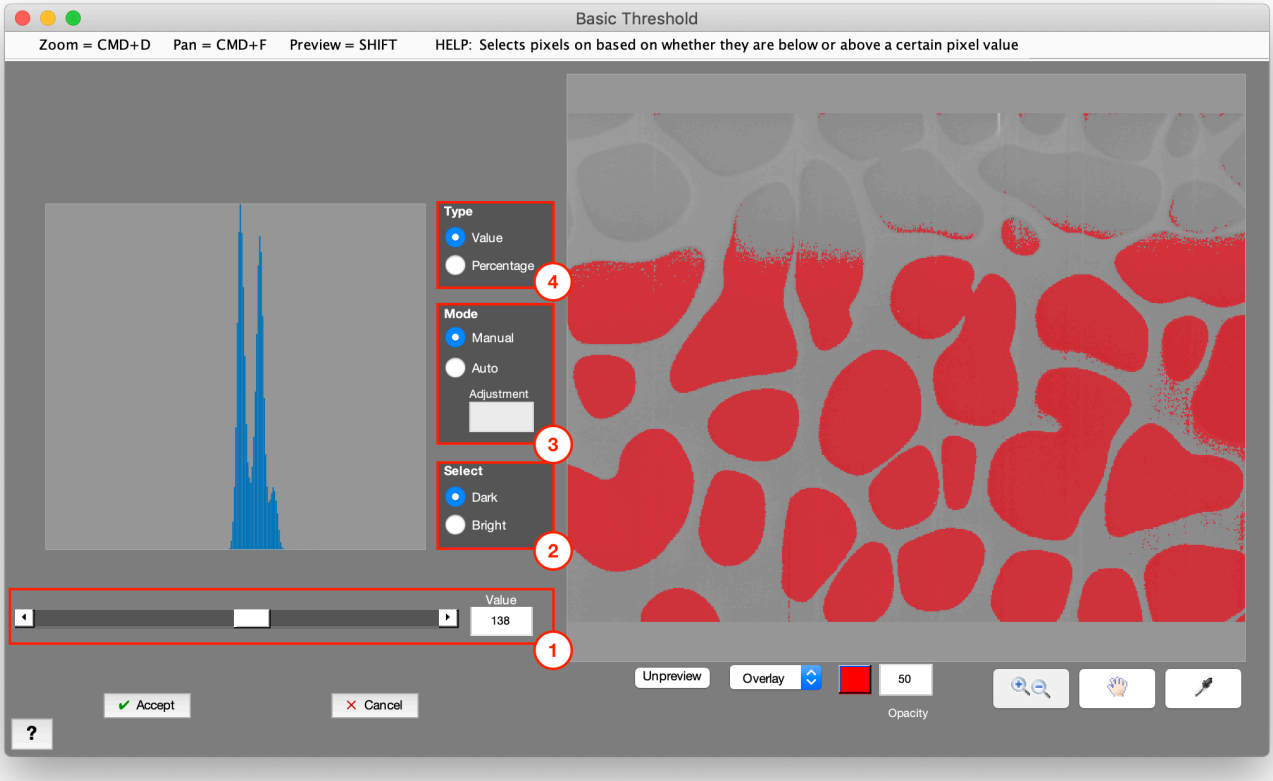
6. Recipe Execution Options

Sets the behavior of the Manual Edit step when a recipe is run in the Image, Batch, and Real-Time Processors. “Reset on Load” clears manual edits each time the recipe is applied to a new image (or re-loaded on an image in the Image Processor). “Interruptible” causes recipe execution to pause and open the Manual Edit tool, allowing the user to make modifications to the segmentation before resuming.

Basic Threshold

Segmentation > Basic Threshold

Selects pixels based on whether they are below or above a certain pixel value. “Auto” determines this threshold value using Otsu’s method, which chooses the value which minimizes the average grayscale variance of the pixels which have been selected and not selected.



1. Value

Threshold value which sets pixels as selected or not

2. Select

- **Dark:** Selects pixels whose values are below the threshold value
- **Bright:** Selects pixels whose values are above the threshold value

3. Mode

- **Manual:** Allows manual selection of threshold value
- **Auto:** Auto-determines threshold value based on Otsu’s method [1]
  - **Adjustment:** Specifies offset to make to auto-determined threshold value.

4. Select

- **Value:** Selects pixels whose values are above or below the threshold value
- **Percentage:** Selects pixels whose values are above or below a percentage of the 0-255 image range

References

[1] Otsu, N., “A Threshold Selection Method from Gray-Level Histograms,” IEEE Transactions on Systems, Man, and Cybernetics, Vol. 9, No. 1, 1979, pp. 62-66.

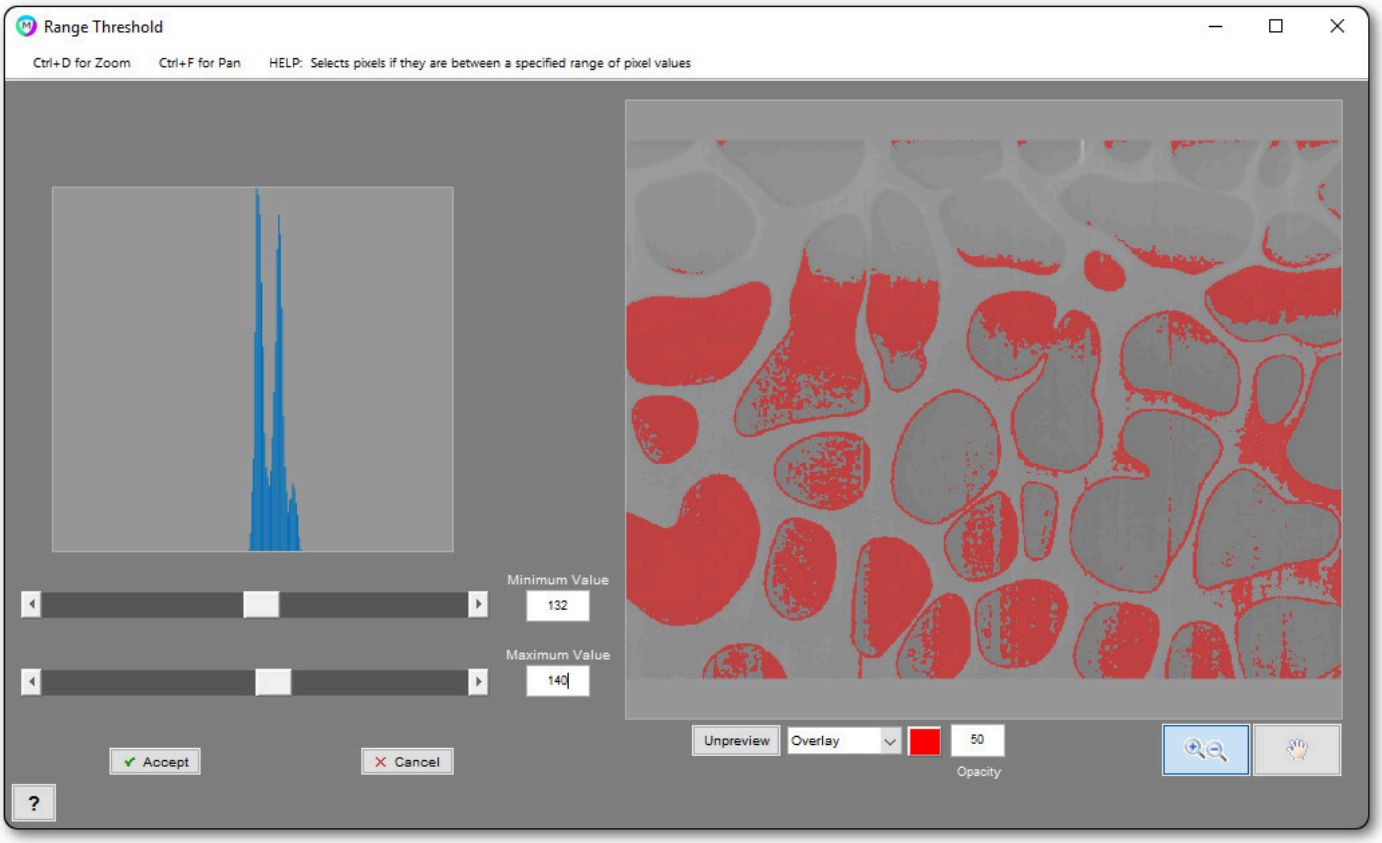
Tips

- Avoid using a Basic Threshold without any pre-processing. This often results in subjective, inconsistent results.
- Recommend trying a Smart Cluster or other image normalization technique before applying a threshold.

Range Threshold

Segmentation > Range Threshold

Selects pixels if they are between a specified range of pixel values.



1. Minimum Value

Minimum threshold value

2. Maximum Value

Maximum threshold value

Tips

- Avoid using a Range Threshold without any pre-processing. This often results in subjective, inconsistent results.
- It is recommended to try a Smart Cluster or other image-normalization technique before applying a threshold.

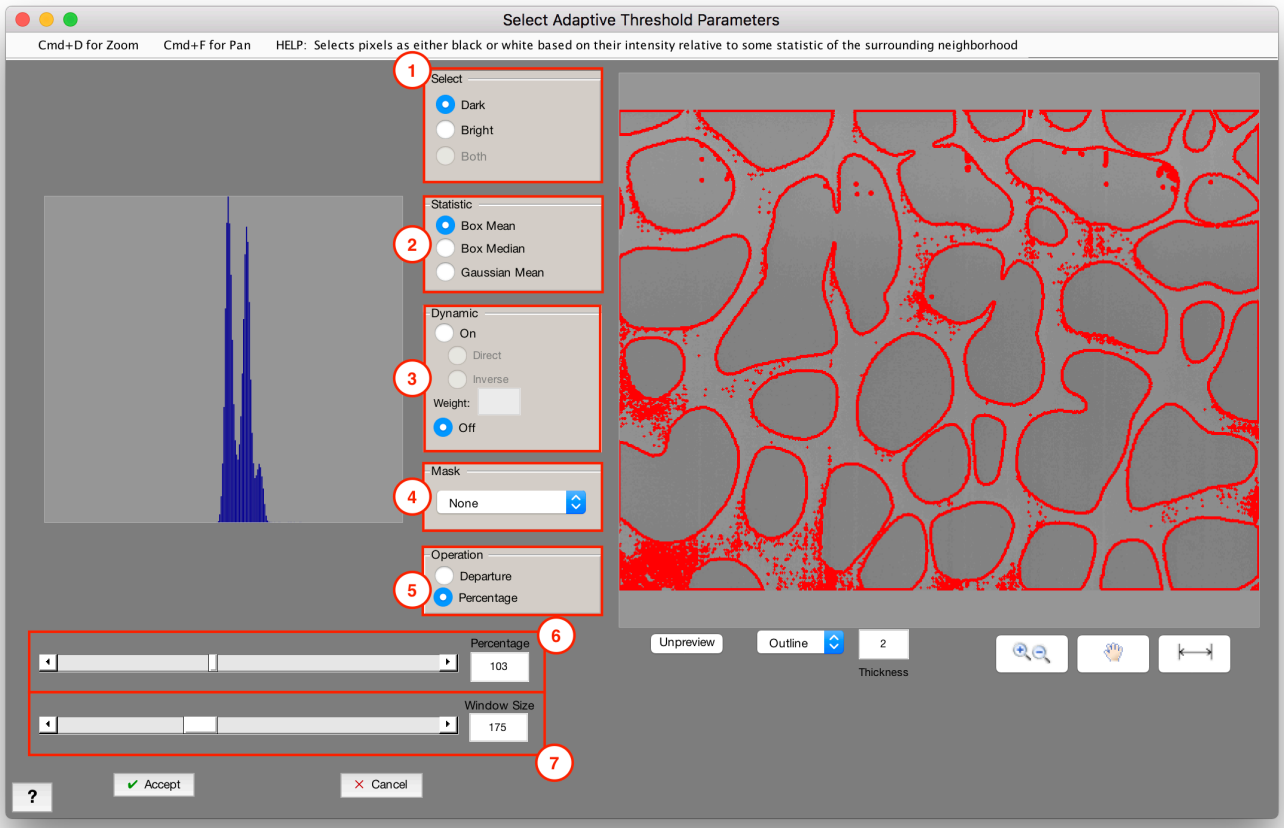
Adaptive Threshold

Segmentation > Adaptive Threshold

Selects pixels based on their intensity relative to some statistic of the surrounding neighborhood.

In the common Percentage mode, with “Statistic” set to Box Mean, pixel values are compared to their local average grayscale value. For example, if Percentage and Box Mean modes are checked, “Select” is set to Dark, Percentage value is 98, and Window Size is 30, then if a pixel is less than 98% of the average grayscale value in a 30×30 pixel window centered around it, it will be selected. If “Select” is set to Bright, then the pixel must be greater than 98% of its local 30×30 window average to be selected.

The window size is typically best set to be just larger than the size of the largest feature one is hoping to capture. The Distance Line tool can help approximate this size.



1. Select

- **Dark:** Selects pixels which meet or fall below the criteria
- **Bright:** Selects pixels which meet or exceed the criteria
- **Both:** Selects pixels that are the + / – the “Departure” value, relative to their neighborhood statistic.

Only applies to “Departure” mode.

2. Statistic

- **Box Mean:** Critical stat for each pixel is the mean of its neighborhood
- **Box Median:** Critical stat for each pixel is the median of its neighborhood
- **Gaussian Mean:** Critical stat for each pixel is the Gaussian mean of its neighborhood

3. Dynamic

- **On:** Critical difference or percentage for each pixel will be scaled based on local variance
  - **Direct:** Higher local variance will raise critical difference or percentage
  - **Inverse:** Higher local variance will lower critical difference or percentage
- **Weight:** Sets how strong the local variance will scale the critical difference or percentage

4. Mask

Only considers and thresholds pixels within the selection of the mask image

5. Operation

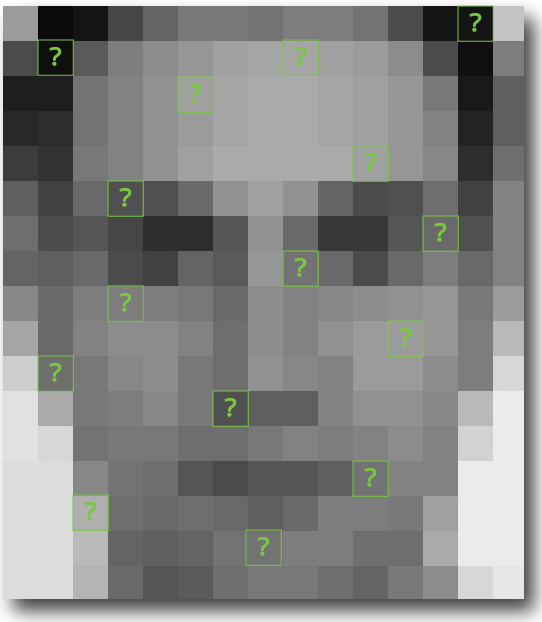
- **Departure:** Sets the critical threshold for each pixel as its difference from its neighborhood statistic
- **Percentage:** Sets the critical threshold for each pixel as its percentage of its neighborhood statistic

6. Percentage/Departure Value

Sets the critical value each pixel get assessed relative to (Recommended: Percentage = 100)

7. Window Size

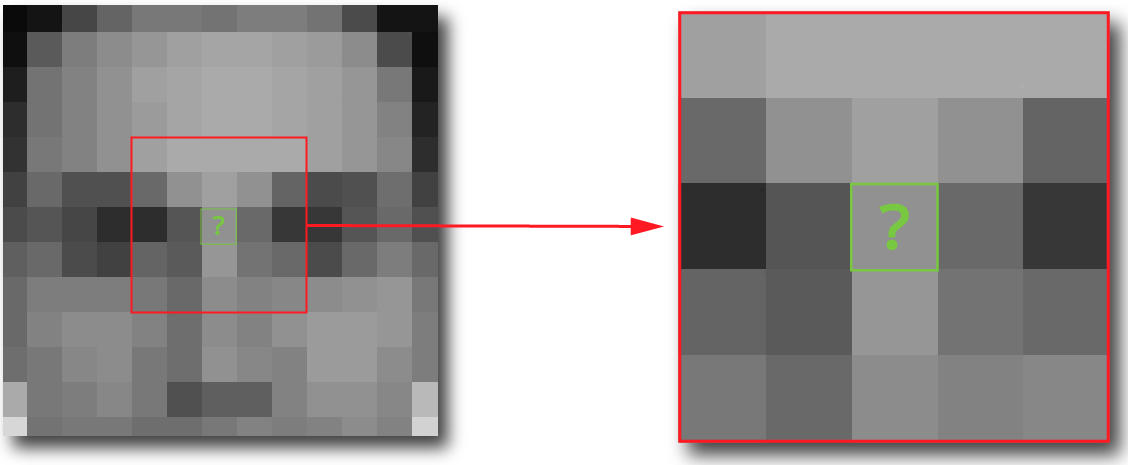
Size of the local neighborhood that each pixel is considered against (Recommended: 30)



Every pixel must be assigned 0 or 1 in order to create a black/white image.

0 - Black

1 - White



A window determined by the “window size” is put around the unknown pixel that is to be calculated.

This window size can be adjusted to include more or less pixels for the calculation.

Before an image is black/white an image is in grayscale. Each pixel in a grayscale image has a grayscale value from 0 to 255.

First, the average grayscale value inside the window is calculated.

Then, the grayscale value of each pixel is compared to the average grayscale value.

Depending on the threshold value, if enough of each pixel’s grayscale value is below or over the average value of the unknown pixel will be assigned 0 or 1.

This kind of calculation is then treated to every pixel.

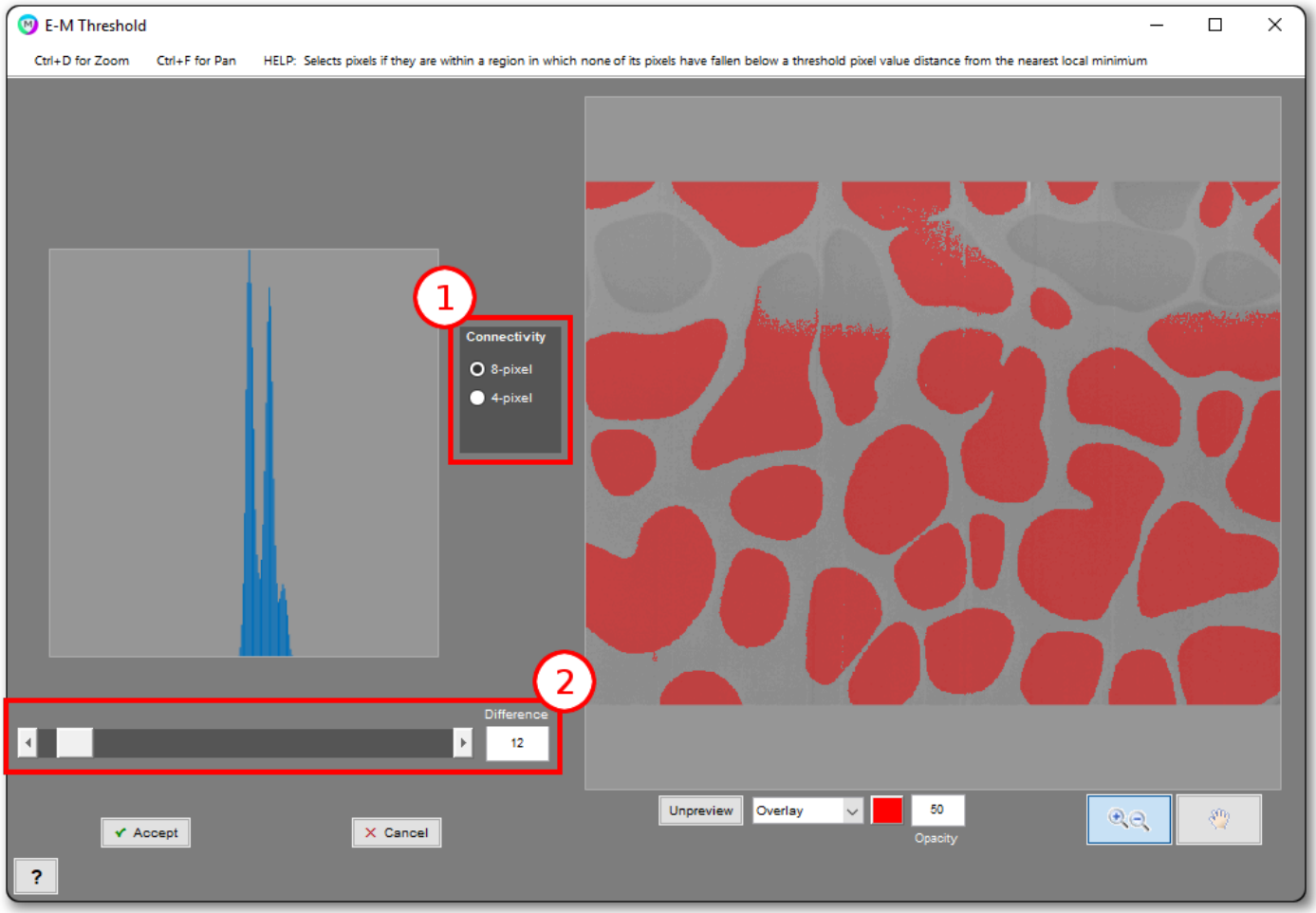
Tips

- To determine a good starting window, using the line tool do some rough measurements of the features you are interested in thresholding
- Starting Window Size Dark Selection: 90, reduce to reduce selection, increase to increase selection. Bright Selection: 120, reduce to increase selection, increase to reduce selection.
- You can limit the threshold to a region of interest set by a Companion mask.

E-M Threshold

Segmentation > E-M Threshold

Selects pixels if they are within a region in which none of its pixel have fallen below a threshold pixel value distance from the nearest local minimum.



1. Connectivity

- **8-pixel:** Uses 8-pixel connectivity (face-based) to determine local minima as the basis for the E-M threshold
- **4-pixel:** Uses 4-pixel (face+corner-based) connectivity to determine local minima as the basis for the E-M threshold

2. Difference

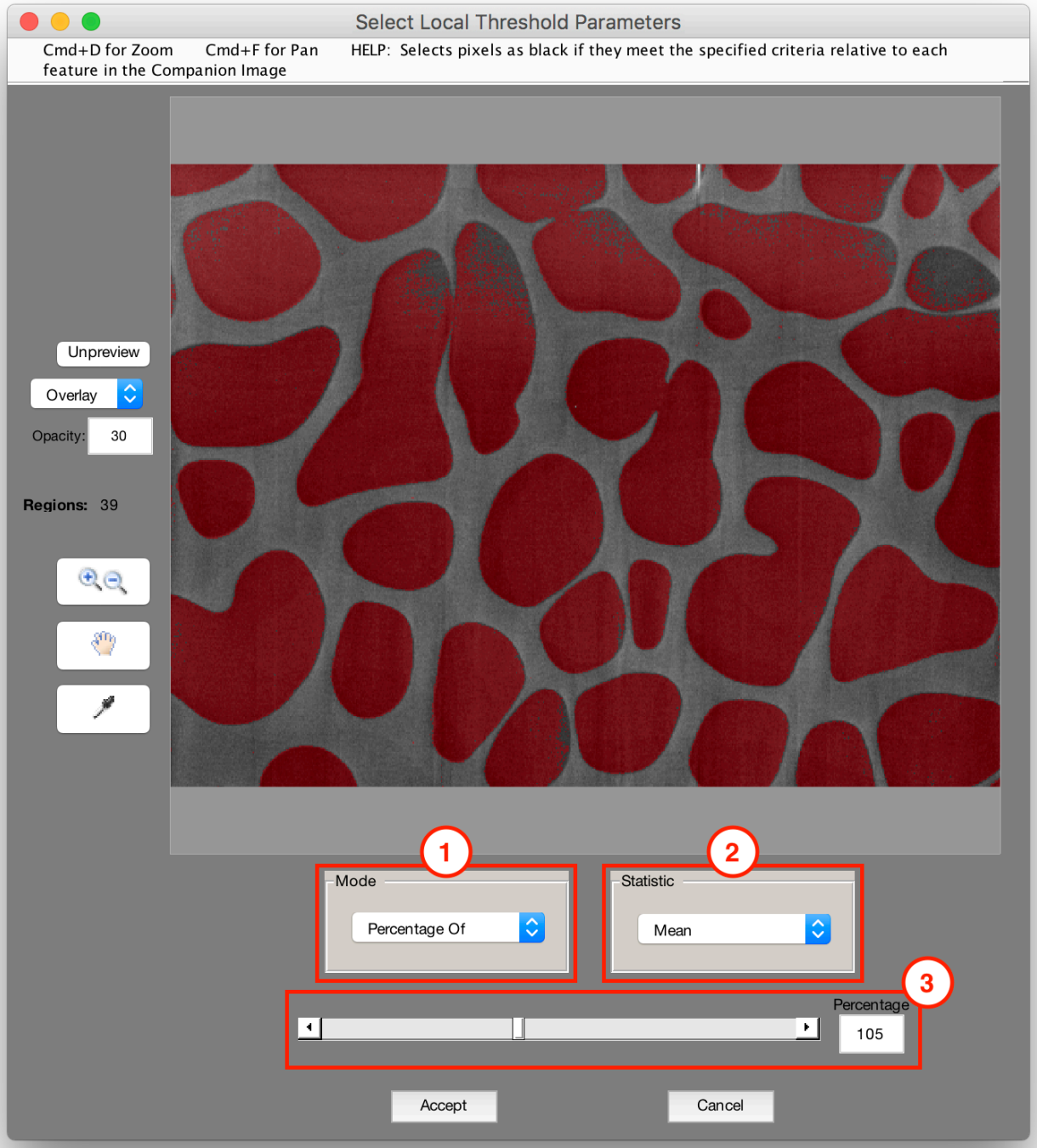
Sets the maximum difference threshold pixel intensities can be from the nearest local minima in order to grouped into that minima's region

*Local Threshold

Segmentation > *Local Threshold

Requires B/W Companion Image

Selects pixels if they meet the specified criteria relative to each feature in the Companion Image. “Auto Threshold” mode calculates the Otsu-based optimum threshold for each feature. “Percentage Of” and “Departure From” asses each pixel relative to its feature’s mean or median pixel value. “Upper Percentile and “Lower Percentile” select the pixels within the specified percentile of their feature’s mean or median pixel value.



1. Mode

Selects which mode to use for local thresholding

- **Auto Threshold:** Selects pixels if they are below the auto-threshold value for each region determined using Otsu's method [1]
- **Percentage Of:** Selects pixels if they are below a certain percentage of their region's critical statistic

- **Difference Below:** Selects pixels if they are below a certain difference from their region's critical statistic
- **Lower Percentile:** Selects pixels if they are with in the specified lower percentile of their region's grayscale pixel values
- **Upper Percentile:** Selects pixels if they are with in the specified upper percentile of their region's grayscale pixel values

2. Statistic

Selects which statistic to consider for the current mode

- **Mean (for Percentage Of and Difference Below):** Compares pixels to their region's mean grayscale intensity
- **Median (for Percentage Of and Difference Below):** Compares pixels to their region's median grayscale intensity

3. Value

- **Adjustment (for Auto Threshold):** Offsets each region's auto-determined threshold value by this amount
- **Percentage (for Percentage Of):** Specifies the critical percentage of the region's mean or median each pixel must be below
- **Difference (for Difference Below):** Specifies the critical difference from the region's mean or median each pixel must be below
- **Percentile (for Lower Percentile):** Specifies the lower percentile of the region's grayscale values that pixels must be within
- **Percentile (for Upper Percentile):** Specifies the upper percentile of the region's grayscale values that pixels must be within

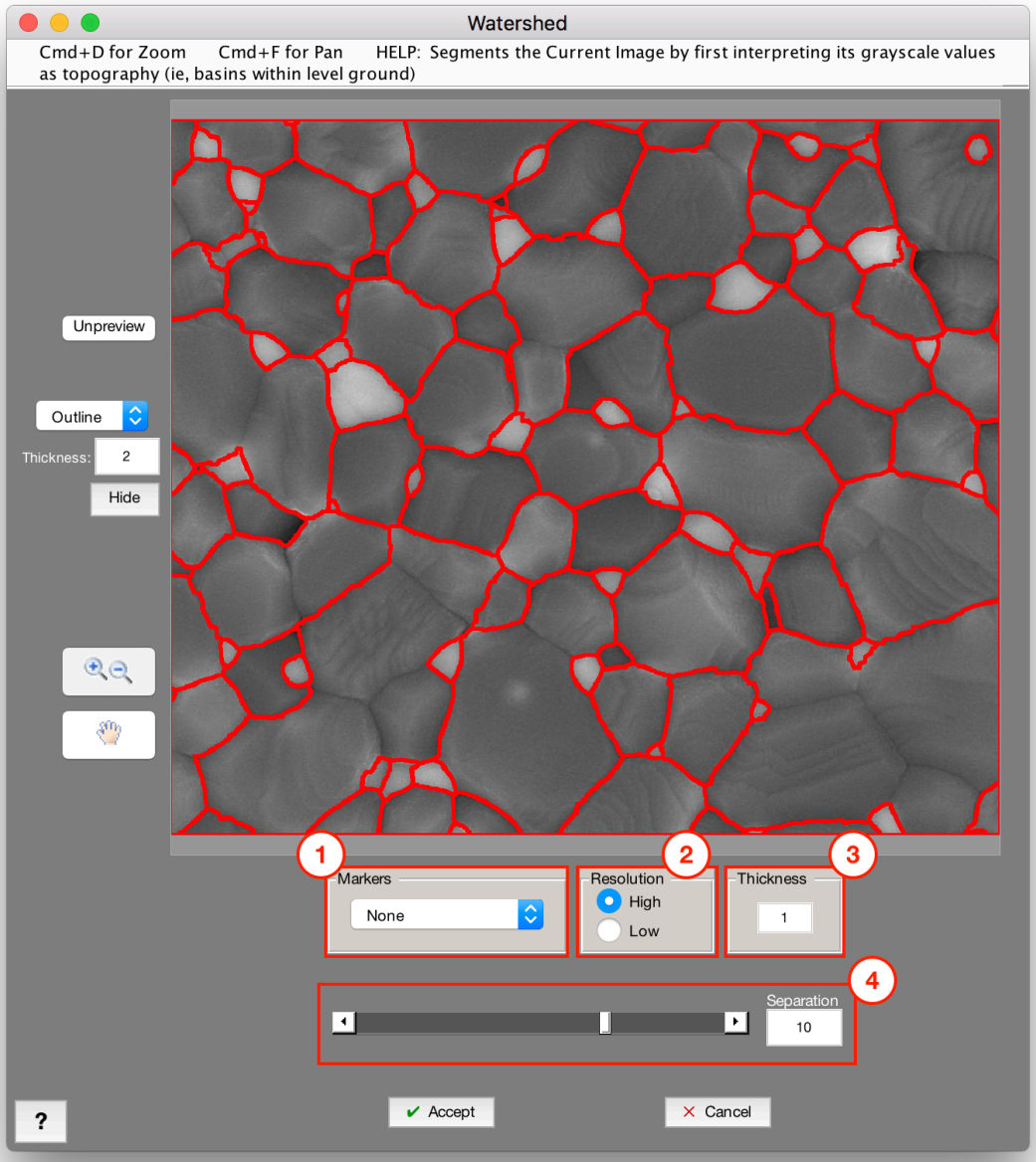
References

[1] Otsu, N., "A Threshold Selection Method from Gray-Level Histograms," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 9, No. 1, 1979, pp. 62-66.

Watershed

Segmentation > Watershed

Segments the Current Image by first interpreting its grayscale values as topography (ie, basins within level ground). A rainfall simulation is then run such that areas of similar intensity are separated from others, where their basins fill to meet [1].



1. Markers

Helps confine feature outlines near your marked locations. Choose to use the selection in a Memory Image

as markers to control watershed. Forces “basins” to be formed at the marker locations prior to the watershed algorithm being run.

2. Resolution

- **High:** Works better for separating features with high pixel density
- **Low:** Works better for separating features with low pixel density (more pixelated)

3. Thickness

Thickness of watershed lines

4. Separation

Controls strength of feature detection. Increase slider to create more feature outlines.

Tips

- Powerful algorithm that can be used in combination with Pre-Processing and Find Texture filters to identify feature boundaries
- When trying to separate already segmented features, use the Separate Features tool instead

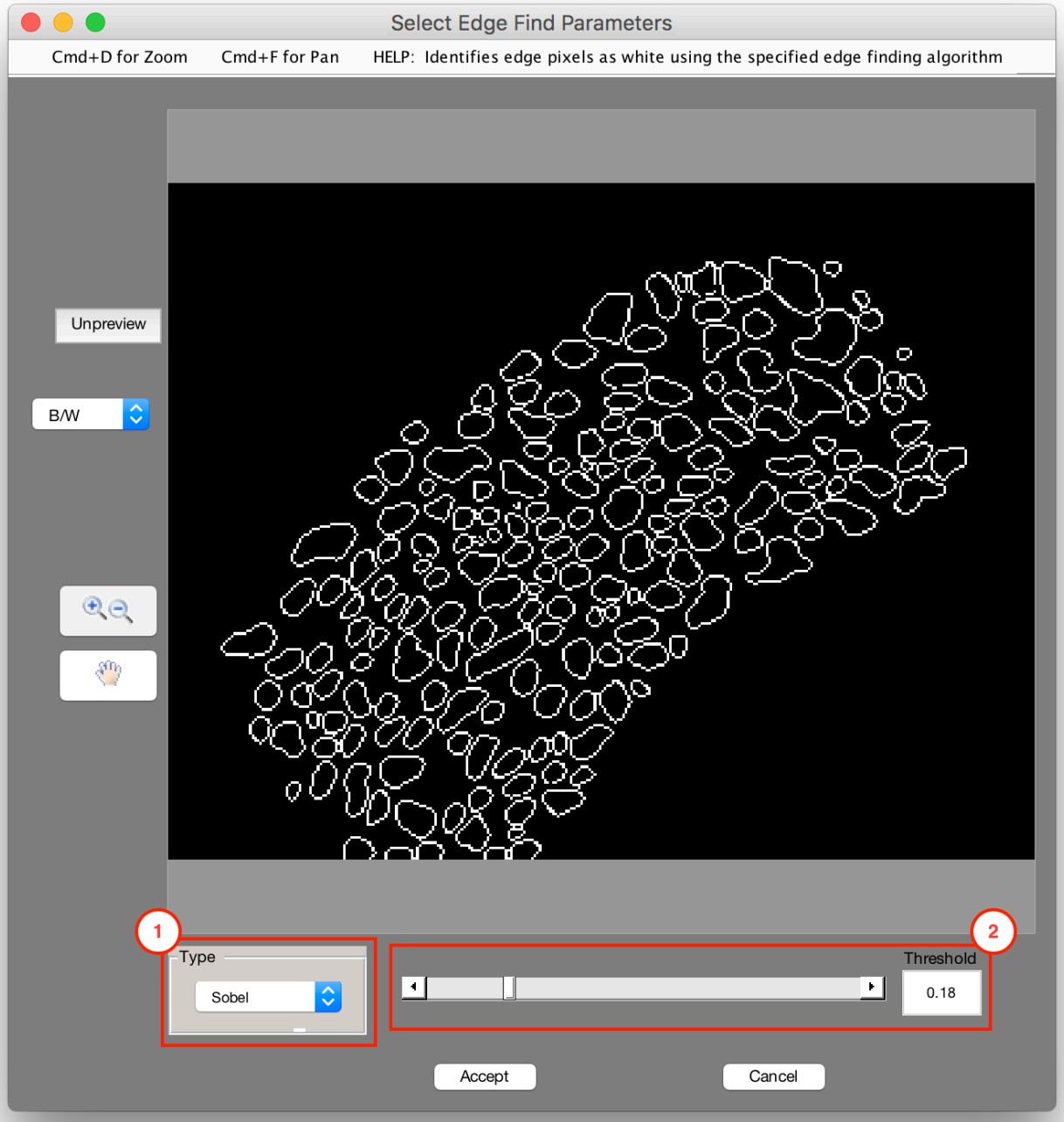
References

[1] Meyer, Fernand, “Topographic distance and watershed lines,” Signal Processing , Vol. 38, July 1994, pp. 113-125.

Find Edges

Segmentation > Find Edges

Sets edge pixels as empty using the specified edge finding algorithm [1-3].



1. Type

The edge finding algorithm used

- **Sobel:** Detect edges using the Sobel algorithm
- **Prewitt:** Detect edges using the Prewitt algorithm
- **Roberts:** Detect edges using the Roberts algorithm
- **Laplacian:** Detect edges using the Laplacian algorithm
- **Zero-Cross:** Detect edges using the Zero-Cross algorithm
- **Canny:** Detect edges using the Canny algorithm (tends to cleaner edges than other algorithms) [3]

2. Threshold

A weakness setting for the edge find algorithm. Higher thresholds are weaker edge finds, but maybe produce less noise. (Recommended: 0.01)

Tips

- It is often useful to optimize the Find Edges function to a Standard Deviation Filter companion image. This allows for dynamic edge finding in presence of variation.

Referrences

[1] Canny, John, “A Computational Approach to Edge Detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence,Vol. PAMI-8, No. 6, 1986, pp. 679-698.

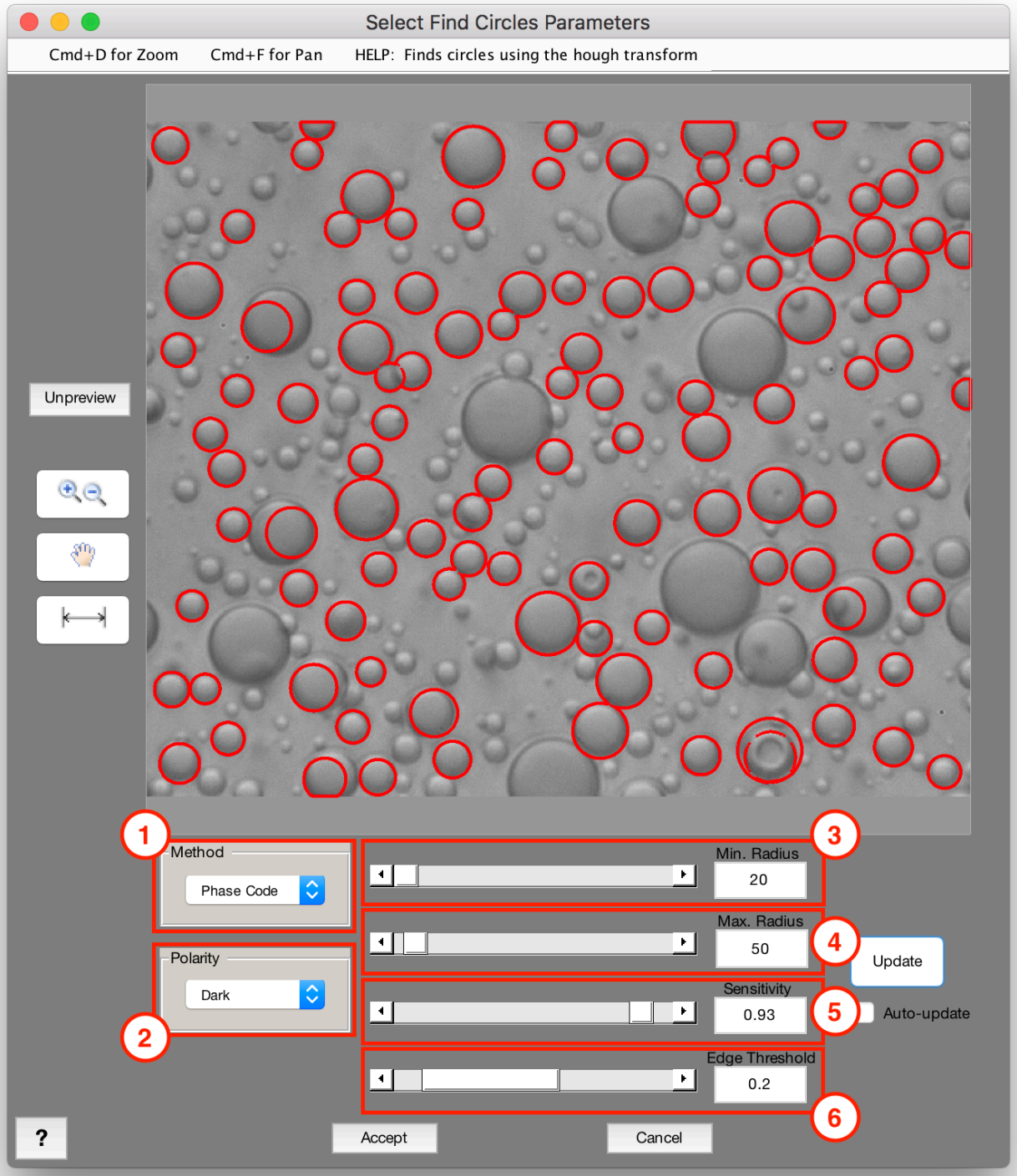
[2] Lim, Jae S., Two-Dimensional Signal and Image Processing, Englewood Cliffs, NJ, Prentice Hall, 1990, pp. 478-488.

[3] Parker, James R., Algorithms for Image Processing and Computer Vision, New York, John Wiley & Sons, Inc., 1997, pp. 23-29.

Find Circles

Segmentation > Find Circles

Finds circles in the Current Image using an algorithm based on searching within the image's Hough transform. This function can take some time to complete depending on the size of and number of circles in the image.



1. Method

Sets method for circle finding. Phase Code tends to be faster and more accurate.

- **Phase-Code:** Uses the phase-code algorithm for circle detection [1]

- **Two-Stage:** Uses the two-stage algorithm for circle detection [2,3]

2. Polarity

Sets whether circle edges are bright or dark outlines in the image

- **Bright:** Looks for circles with bright outlines
- **Dark:** Looks for circles with dark outlines

3. Min. Radius

Minimum radius of circles to be found (Recommended: 10-20)

4. Max. Radius

Maximum radius of circles of the found (Recommended: 20 larger than Min. Radius)

5. Sensitivity

Sets how much contrast needs to be between circles and the background. A higher sensitivity will find more circles, but may increase false positives.

6. Edge Threshold

Another parameter which affects how many circles are found. A lower Edge Threshold finds more circles, but may increase false positives.

Tips

- For best performance we recommend spacing the min and max diameter by 20.
- If a max-min spacing of 20 is too narrow for your application, try running *Find Circles* over multiple iterations to cover the necessary range, and add the results together using *Set Companion Image* and *Union*.

References

[1] T.J Atherton, D.J. Kerbyson. “Size invariant circle detection.” Image and Vision Computing. Volume 17, Number 11, 1999, pp. 795-803.

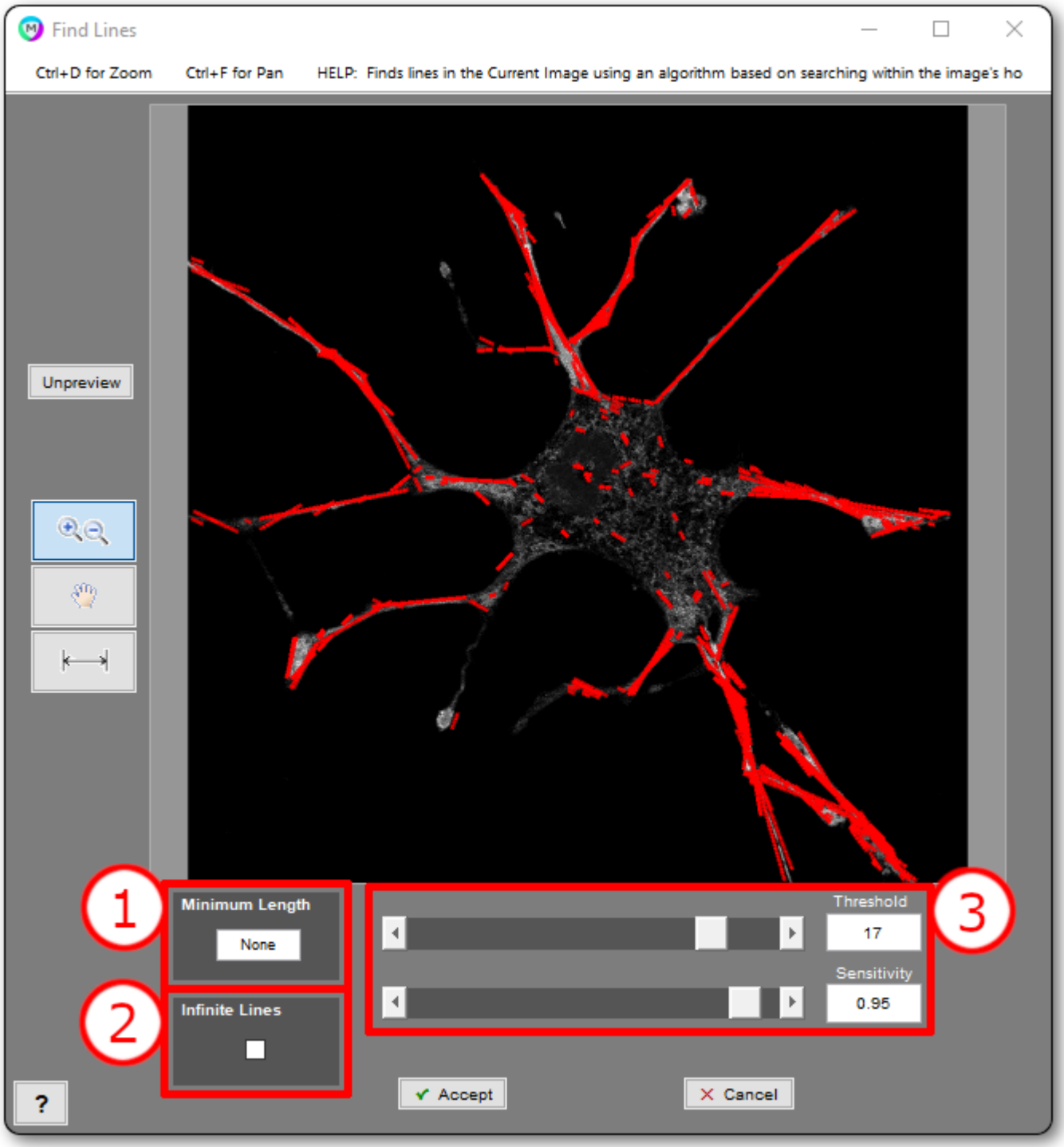
[2] H.K Yuen, .J. Princen, J. Illingworth, and J. Kittler. “Comparative study of Hough transform methods for circle finding.” Image and Vision Computing. Volume 8, Number 1, 1990, pp. 71–77.

[3] E.R. Davies, Machine Vision: Theory, Algorithms, Practicalities. Chapter 10. 3rd Edition. Morgan Kauffman Publishers, 2005,

Find Lines

Segmentation > Find Lines

Finds lines in the Current Image using an algorithm based on searching within the image's Hough transform.



1. Minimum Length

Rejects lines under a certain length.

2. Infinite Lines

Allows the detection of lines that proceed to infinity, in addition to line segments.

3. Detection Strength Settings

Threshold determines how strong a feature must be above background to be considered as a candidate line.

Sensitivity determines how eager the algorithm is to find lines within the image.

Comments

Line detection is relative to the strongest features in the image. Detection will be much more successful if items such as scale bars and information boxes are removed from the image first.

Advanced

Contains advanced functions to recognize and select different types of features.

Functions

- Find Text
- Find Facial Features

Find Text

Segmentation > Advanced > Find Text

Finds text in the Current Image using optical character recognition (OCR) [1] and places boxes around each detected word.

> See the “Text” item under “Based on Companion” in Feature Measurements for information on how to extract recognized text strings

1. Character Set

Specifies the type of characters searched for

- **All:** Searches for all English characters
- **Numbers:** Searches for numbers only

2. Detection

Controls the minimum acceptable confidence of detected text. Increase the slider to increase the sensitivity of the text detection algorithm. (Recommended: 0.5)

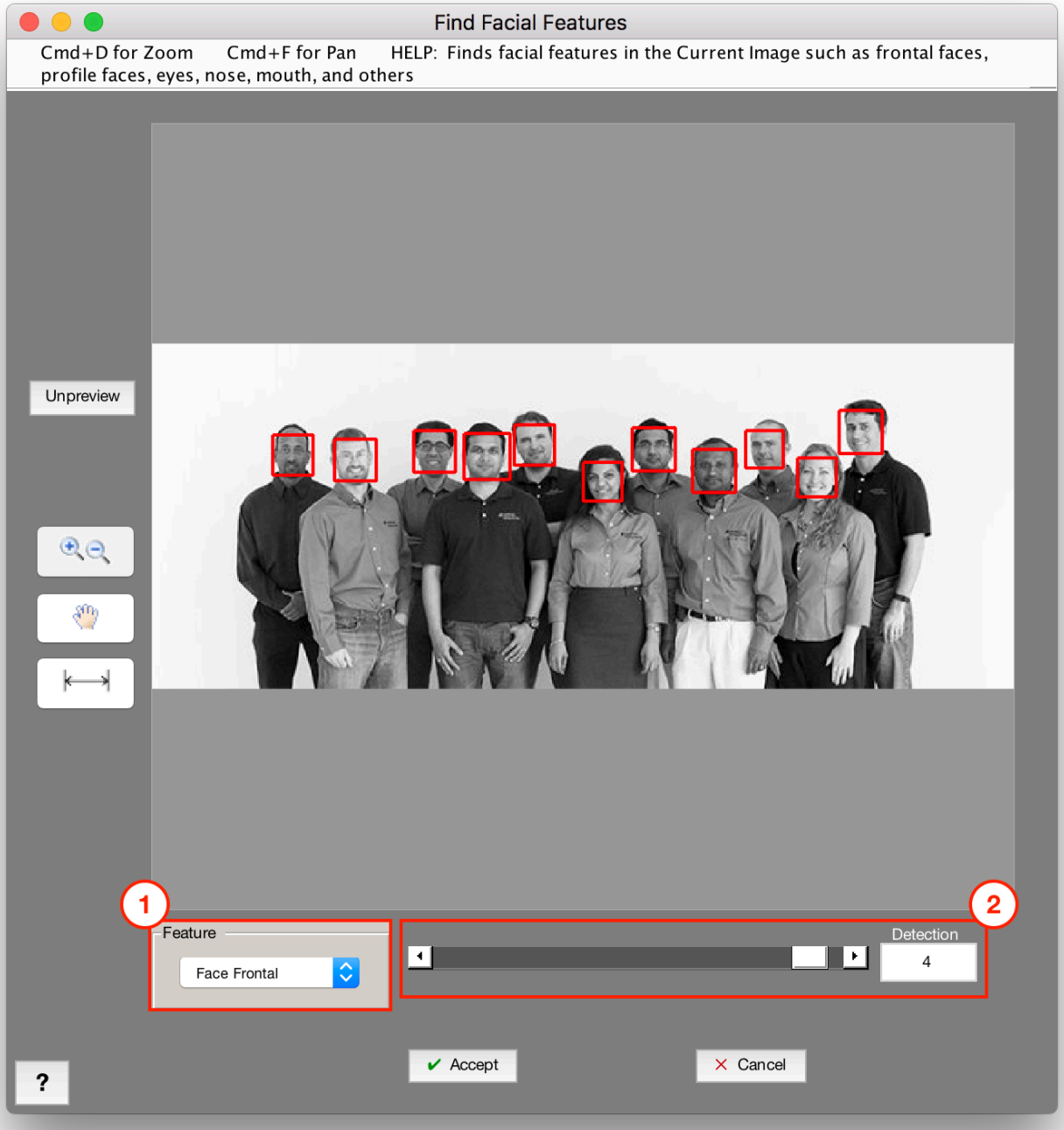
References

[1] R. Smith. An Overview of the Tesseract OCR Engine, Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007) Vol 2 (2007), pp. 629-633.

Find Facial Features

Segmentation > Advanced > Find Facial Features

Finds facial features in the Current Image such as frontal faces, profile faces, eyes, nose, mouth, and others.



1. Feature

Sets type of facial feature to detect

- **Face Frontal:** Detects front-facing faces using CART-based classifiers [1]
- **Face Profile:** Detects upright face profiles using Haar features passed through a decision stump
- **Eye Pair (small):** Detects pairs of eyes. Better suited for smaller eye pairs [2].
- **Eye Pair (large):** Detects pairs of eyes. Better suited for larger eye pairs [2].
- **Left Eye:** Detects left eye using Haar features passed through a decision stump [2]
- **Right Eye:** Detects right eye using Haar features passed through a decision stump [2]
- **Nose:** Detects nose using Haar features passed through a decision stump [2]
- **Mouth:** Detects mouth using Haar features passed through a decision stump [2]
- **Upper Body:** Detects the upper body (i.e., head and shoulders area) [3]
- **Pedestrians 1 (small):** Detects pedestrians at a distance using HOG features [4]
- **Pedestrians 1 (large):** Detects pedestrians close up [4]
- **Pedestrians 2:** Detects pedestrians using ACF features trained on the INRIA Person dataset [5]
- **Pedestrians 3:** Detects pedestrians using ACF features trained on the Caltech Pedestrian dataset [5]

2. Detection

Factor which controls the detection strength of facial feature finding. Increase the slider to increase the sensitivity of feature-finding algorithm. (Recommended: 4-10)

References

[1] Lienhart R., Kuranov A., and V. Pisarevsky “Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection.”, Proceedings of the 25th DAGM Symposium on Pattern Recognition. Magdeburg, Germany, 2003.

[2] Castrillón Marco, Déniz Oscar, Guerra Cayetano, and Hernández Mario, “ENCARA2: Real-time detection of multiple faces at different resolutions in video streams”. In Journal of Visual Communication and Image Representation, 2007 (18) 2: pp. 130-140.

[3] Kruppa H., Castrillon-Santana M., and B. Schiele. “Fast and Robust Face Finding via Local Context”. Proceedings of the Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2003, pp. 157–164.

[4] Dalal, N. and B. Triggs. “Histograms of Oriented Gradients for Human Detection,“Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, June 2005, pp. 886-893.

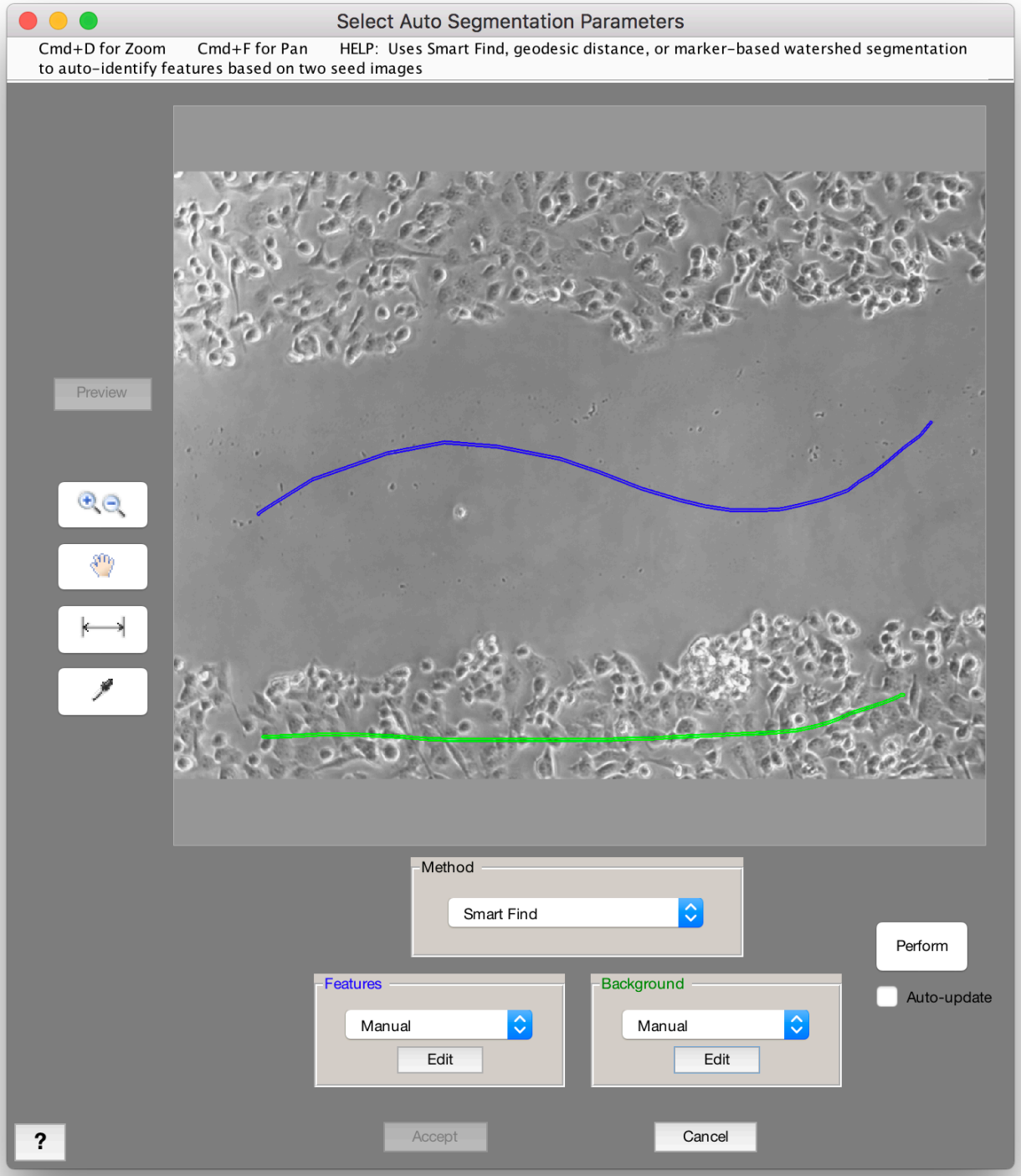
[5] Dollar, C. Wojeck, B. Shiele, and P. Perona. “Pedestrian detection: An evaluation of the state of the art.” Pattern Analysis and Machine Intelligence, IEEE Transactions.Vol. 34, Issue 4, 2012, pp. 743–761.

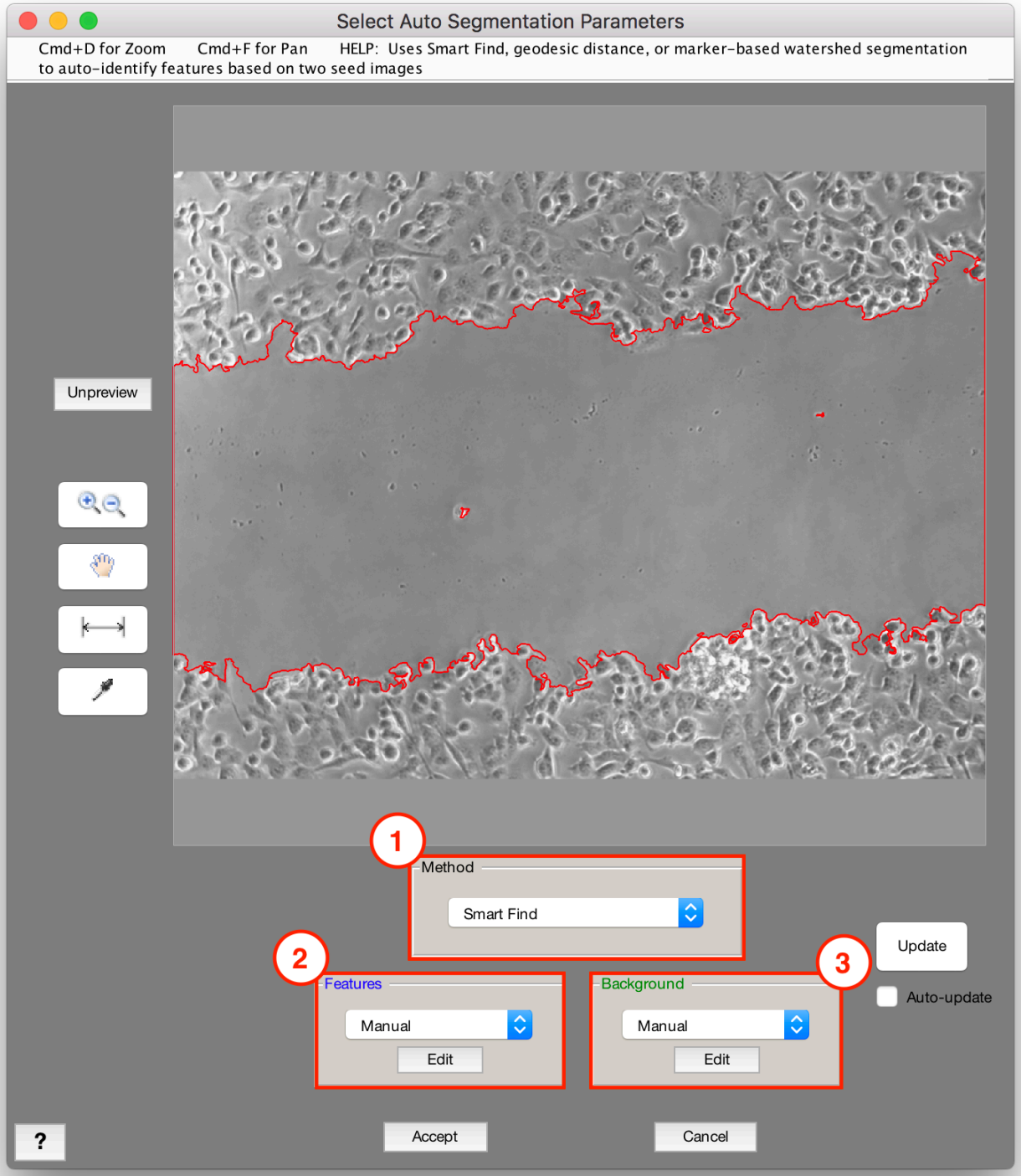
Auto Segmentation

Segmentation > Auto Segmentation

May use up to two B/W memory images

Uses Smart Find™, geodesic distance, or marker-based watershed segmentation to auto-identify features based on two seed images. One seed image roughly identifies features and the other the background. Seed images can be any of the available B/W memory images.


Select Auto Segmentation Parameters
Cmd+D for Zoom Cmd+F for Pan HELP: Uses Smart Find, geodesic distance, or marker-based watershed segmentation to auto-identify features based on two seed images
Preview
Method
Smart Find
Perform
Auto-update
Features
Manual
Edit
Background
Manual
Edit
?
Accept
Cancel



1. Method

- **Smart Find:** Uses a graph cut algorithm [1] to auto-determine feature boundaries in the image based on the feature and background markers. The most effective algorithm for this purpose. Effective even when only “scribbles” have been manually drawn and stored as memory images, and especially

effective when partial selections of some features and background have been made from other Recipe steps.

- **Geodesic Distance:** Uses the geodesic distance method [2] to auto-determine feature boundaries in the image based on the feature and background markers. Useful when only partial selections of nearly all features have been captured and the user needs the software to “snap” the boundaries into place.
- **Marker-Based Watershed:** First forces the image to have local minima at the feature and background marker locations. Then applies the watershed algorithm [3] to auto-determine feature boundaries. Tends to grow the existing feature boundaries about halfway to the background markers.

2. Features

Uses manual tracings, or takes selected (black) features from a memory image, to help identify features for auto segmentation

3. Background

Uses manual tracings, or takes selected (black) features from a memory image, to help identify the background for auto segmentation

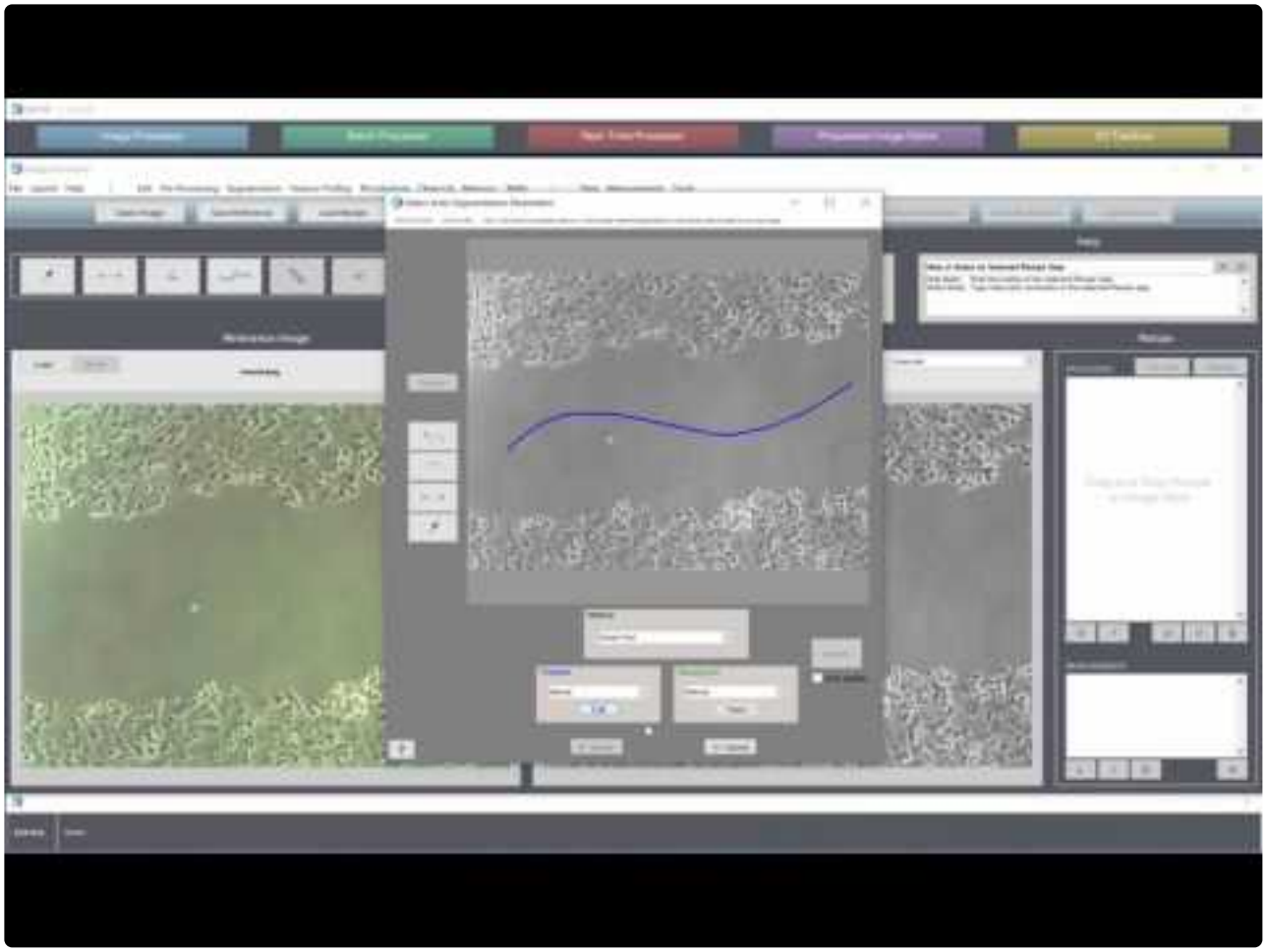
References

[1] Y. Li, S. Jian, C. Tang, H. Shum, Lazy Snapping In Proceedings from the 31st International Conference on Computer Graphics and Interactive Techniques, 2004.

[2] A. Protiere and G. Sapiro, Interactive Image Segmentation via Adaptive Weighted Distances, IEEE Transactions on Image Processing, Volume 16, Issue 4, 2007.

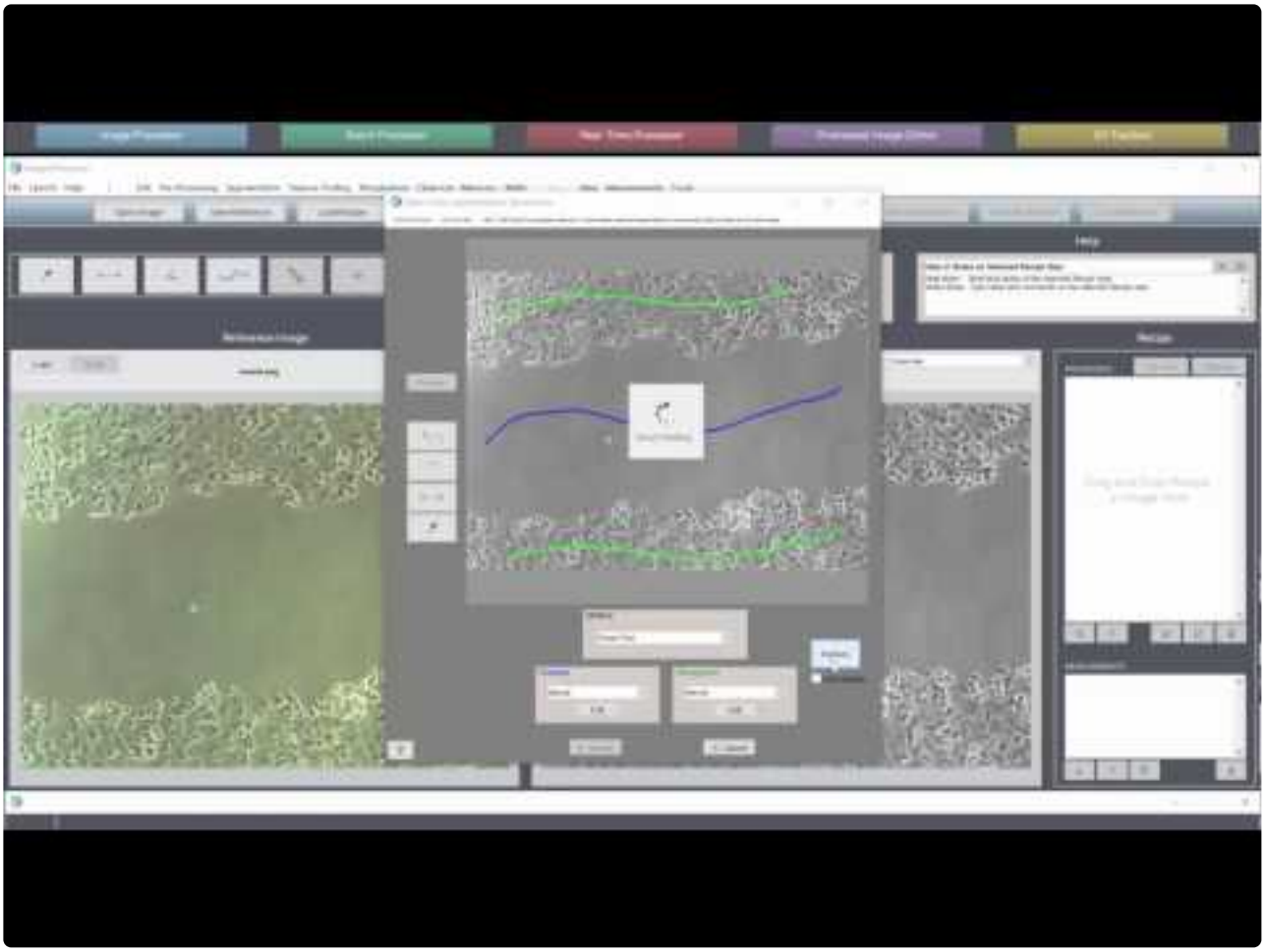
[3] Meyer, Fernand, “Topographic distance and watershed lines,” Signal Processing , Vol. 38, July 1994, pp. 113-125.

Demo



https://www.youtube.com/embed/Fx0G8hCYDes?rel=0

Tutorial



https://www.youtube.com/embed/GSrwRvSFh4E?rel=0

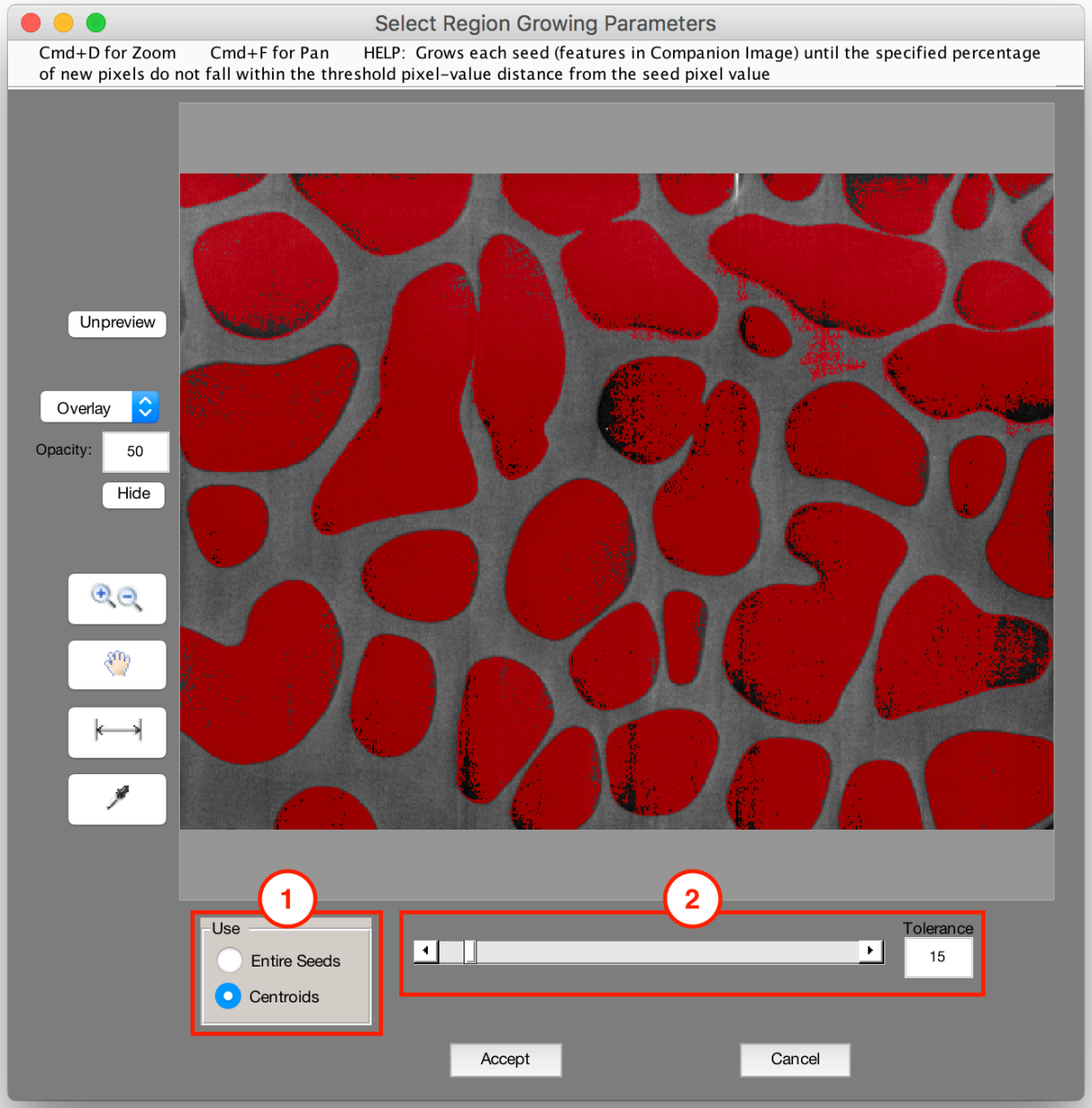
https://www.youtube.com/embed/GSrwRvSFh4E?rel=0

*Region Grow

Segmentation > *Region Grow

Requires B/W Companion Image

Grows each seed (features in Companion Image) until the specified percentage of new pixels do not fall within the threshold pixel-value distance from the seed pixel value. Seeds are are obtained from the features in the Companion Image.



1. Use

- **Entire Seeds:** Use entire seeds for region growing. Entire seed will be superimposed on each region after growing. Region growing will still start from seed centroids.
- **Centroids:** Use centroid of each seed for region growing.

2. Tolerance

Tolerance for region growing. Only pixels within the tolerance threshold of the seed pixel get added to the

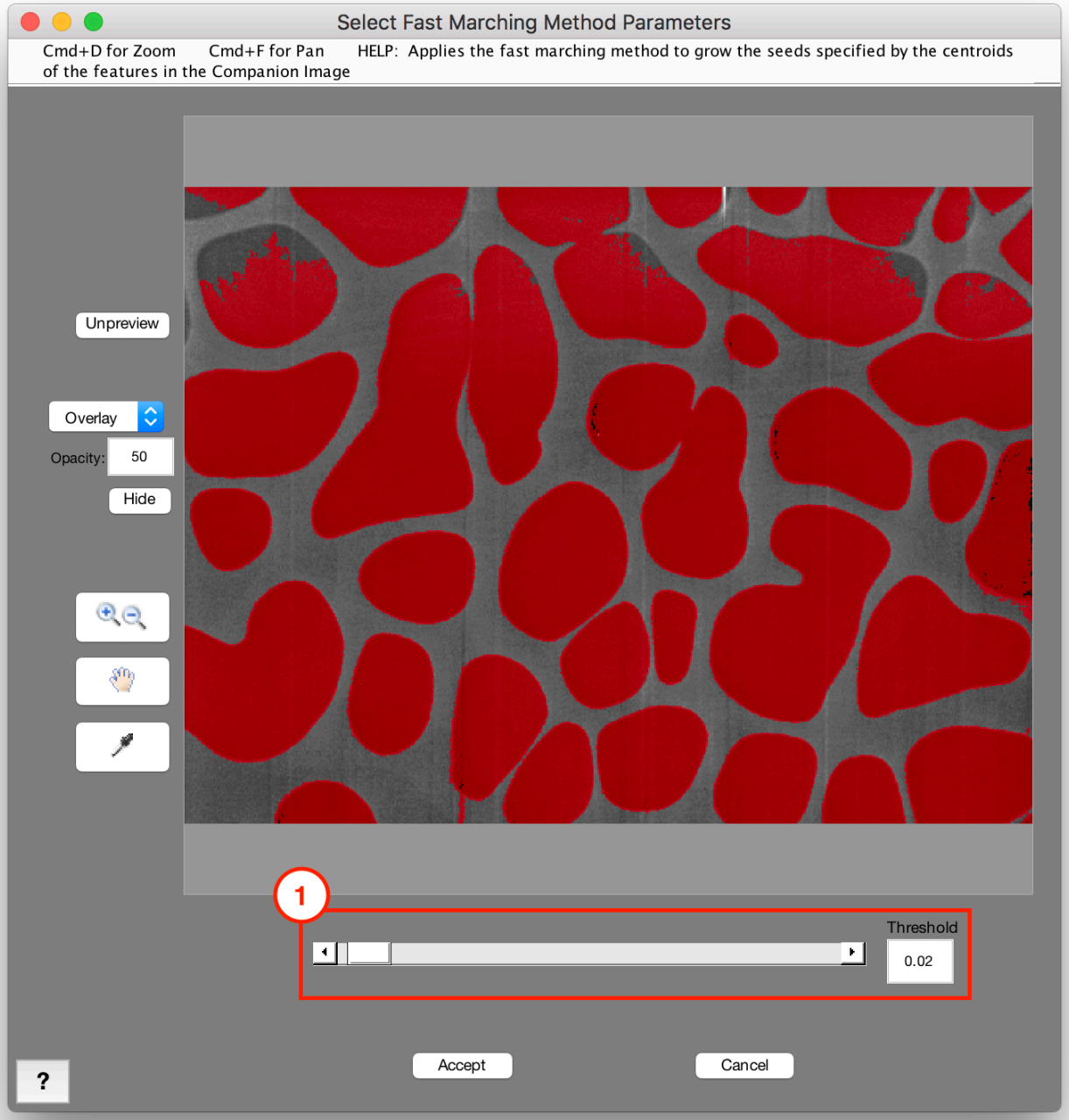
region.

*Fast Marching Method

Segmentation > *Fast Marching Method

Requires B/W Companion Image

Applies the fast marching method [1] to grow the seeds specified by the centroids of the features in the Companion Image. The algorithm is a more robust version of region growing, where regions grow according to the local grayscale difference around them. The seeds mark the locations to first measure grayscale. Then, all pixels are weighted according to the relative grayscale value difference to the nearest seed. This is referred to as the weight map.



1. Threshold

Cutoff value to determine when centroids of features (seeds) stop growing. Larger value allows greater growth. (Recommended: 0.01-0.05)

References

[1] Sethian, J. A. Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational

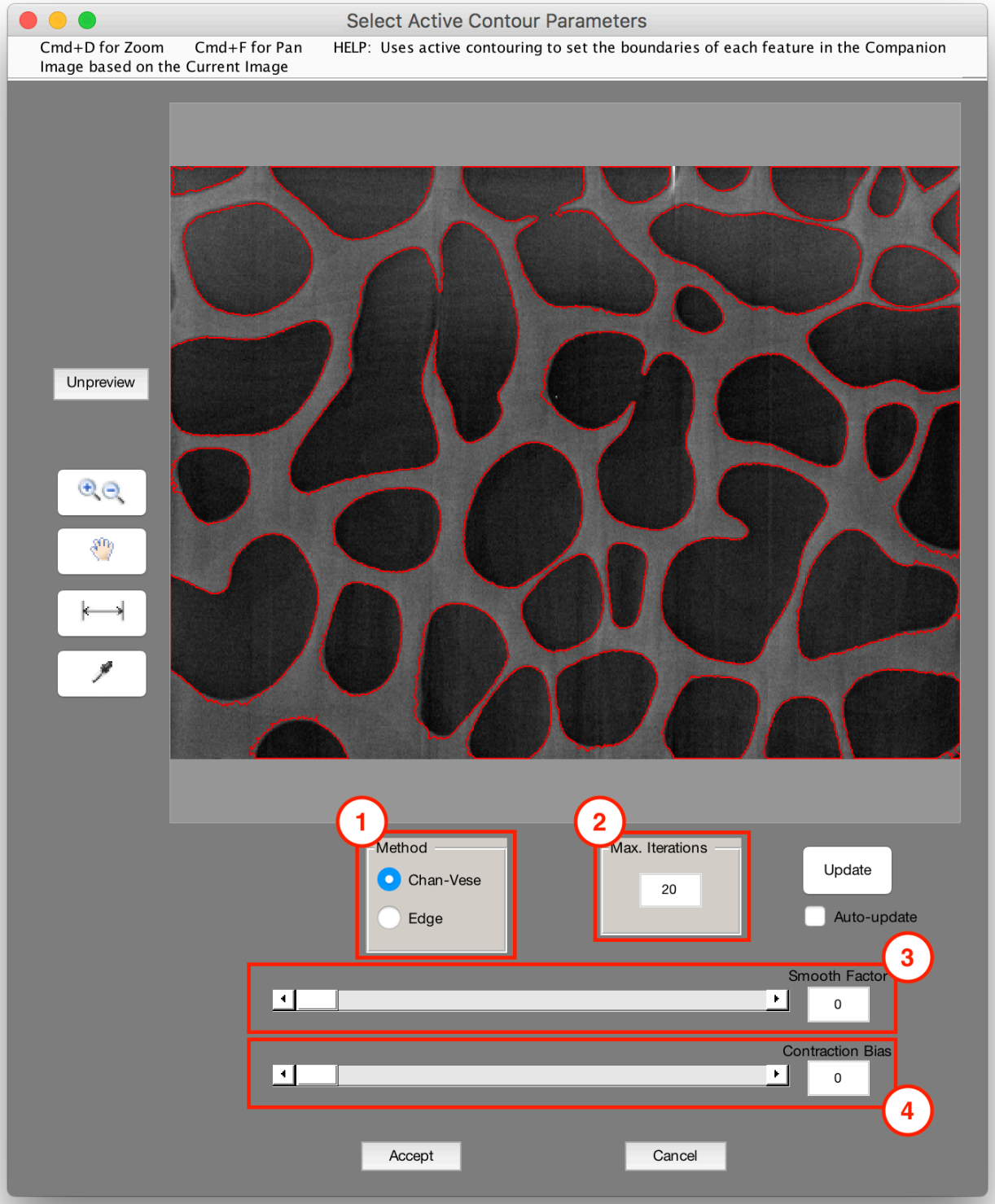
Geometry, Fluid Mechanics, Computer Vision, and Materials Science, Cambridge University Press, 2nd Edition, 1999.

*Active Contour

Segmentation > *Active Contour

Requires B/W Companion Image

Uses active contouring [1,2] to set the boundaries of each feature in the Companion Image based on the Current Image. Active contouring grows or shrinks features to “snap” them to locally high-contrast boundaries. The Contraction Bias allows users to add some external “force” to the contour to increase the likelihood that it will shrink.



1. Method

- **Chan-Vese:** Uses the Chan-Vese method to drive the contour [1]. Tends to work faster than the edge method.

- **Edge:** Uses the edge method to drive the contour [2]. Is naturally more biased to shrink the contour. Tends to work more slowly than the Chan-Vese method.

2. Max. Iterations

The maximum number of iterations to evolve the active contour

3. Smooth Factor

Controls the resulting roughness of the active contour. Higher is smoother.

4. Contraction Bias

Controls the external force on the active contour. Higher makes it tougher for the active contour to grow.

References

[1] T. F. Chan, L. A. Vese, Active contours without edges. IEEE Transactions on Image Processing, Volume 10, Issue 2, pp. 266-277, 2001

[2] V. Caselles, R. Kimmel, G. Sapiro, Geodesic active contours. International Journal of Computer Vision, Volume 22, Issue 1, pp. 61-79, 1997.

Find Global Maximum

Segmentation > Find Global Maximum

Selects the global maximum pixel in the Current Image.

Find Global Minimum

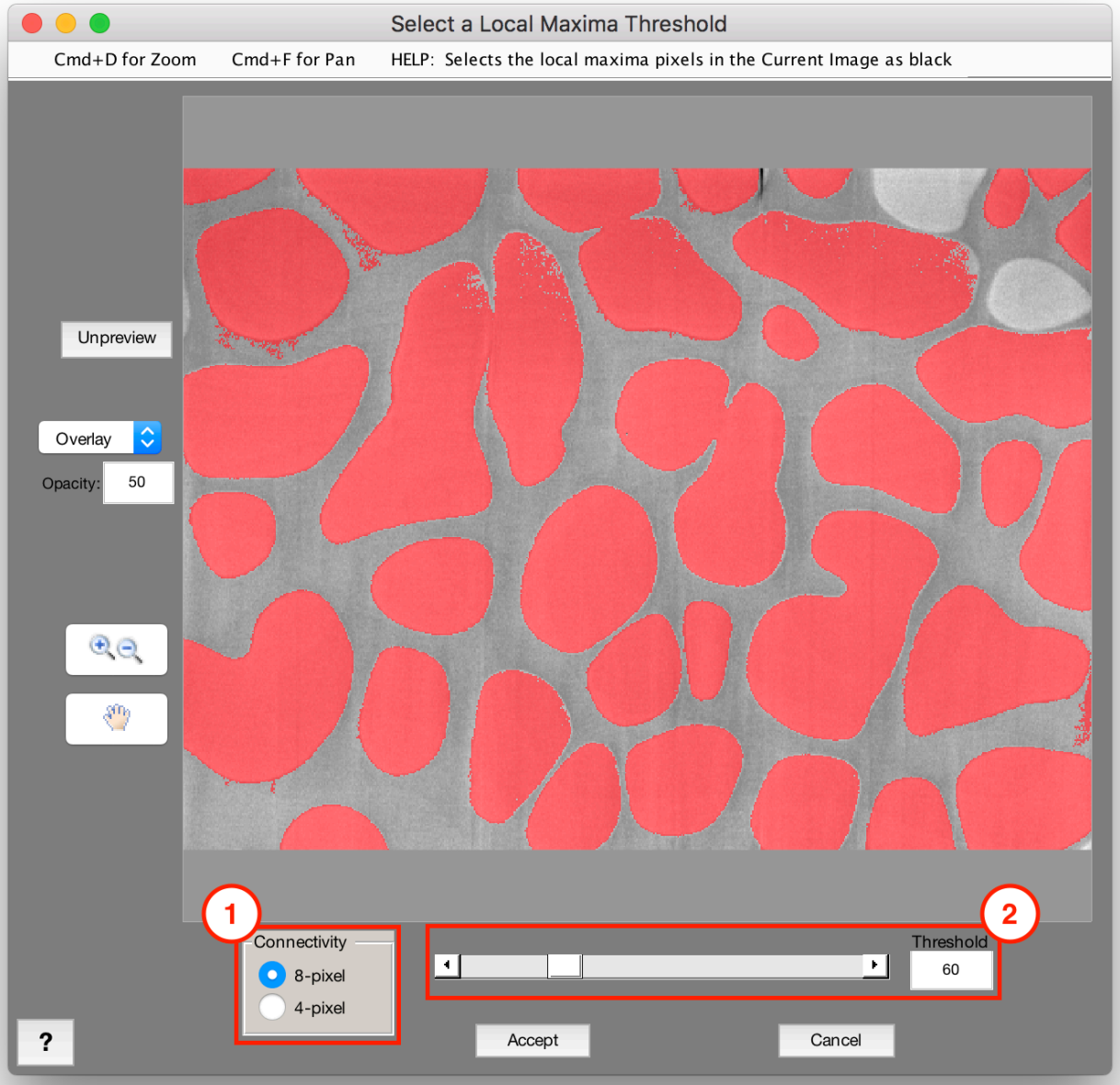
Segmentation > Find Global Minimum

Selects the global minimum pixel in the Current Image.

Find Local Maxima

Segmentation > Find Local Maxima

Selects the local maxima pixels in the Current Image.



1. Connectivity

- **8-pixel:** Uses 8-pixel connectivity to determine local maxima regions
- **4-pixel:** Uses 4-pixel connectivity to determine local maxima regions

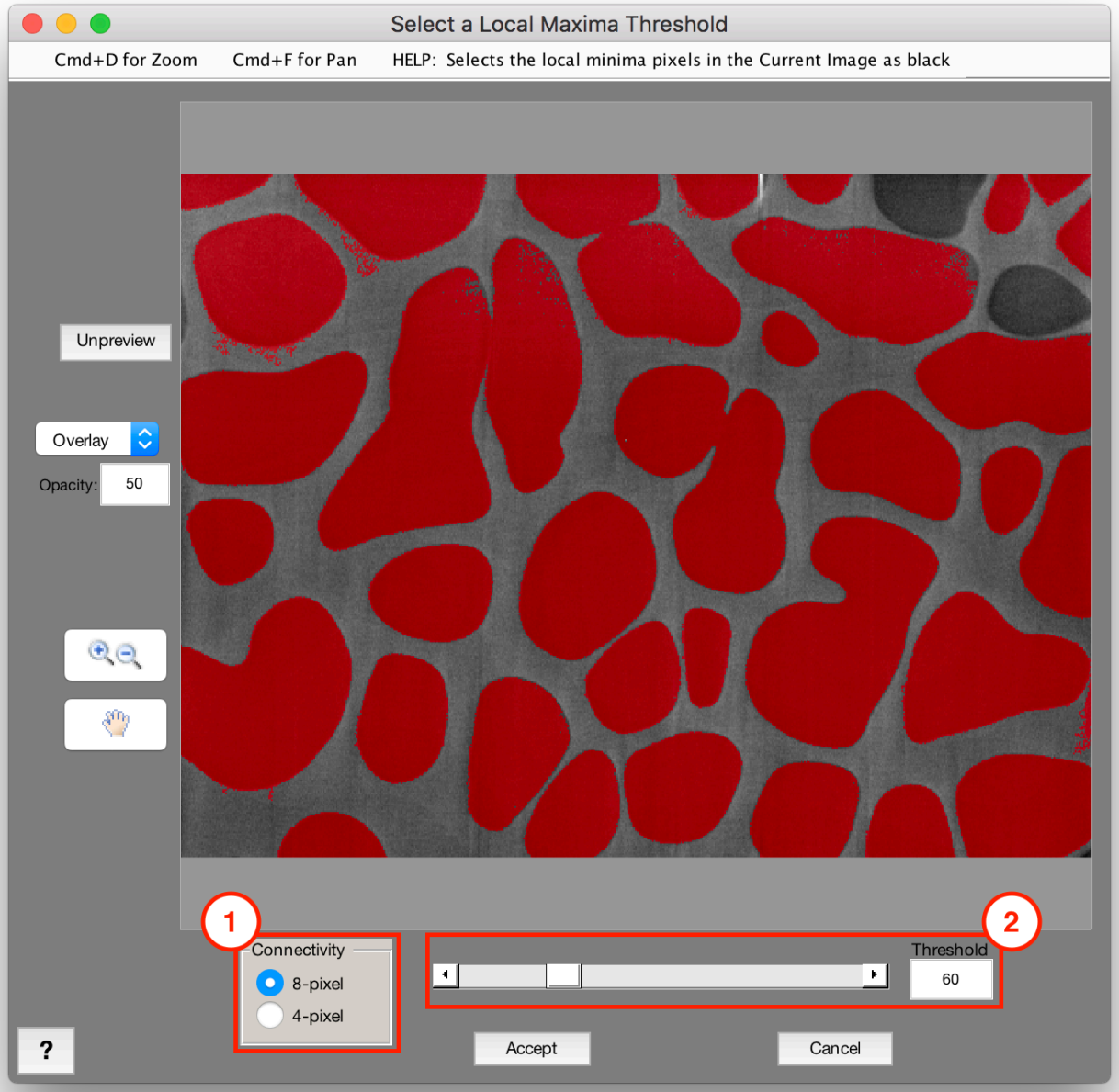
2. Threshold

Ignores local maxima whose intensity is not at least the threshold value above their surroundings

Find Local Minima

Segmentation > Find Local Minima

Selects the local minima pixels in the Current Image.



1. Connectivity

- **8-pixel:** Uses 8-pixel connectivity to determine local minima regions
- **4-pixel:** Uses 4-pixel connectivity to determine local minima regions

2. Threshold

Ignores local minima whose intensity is not at least the threshold value below their surroundings

Morphology

Contains functions for manipulating the shape and size of selected features. These include functions such as dilation, erosion, separation, skeletonization, and shape replacement.

Functions

Dilate and Erode

- Dilate
  - Uniform
  - Smart
  - Retain
- Erode
  - Uniform
  - Smart
  - Retain

Edges

- Separate Features
- Clean Boundaries
- Smooth Features
- Extend Features
- Perimeter Pixels

Thin

- Skeletonization
- Thin
- Shrink
- Distance Map

Dilate

Contains morphological operations which expand a selection mask according to certain parameters.

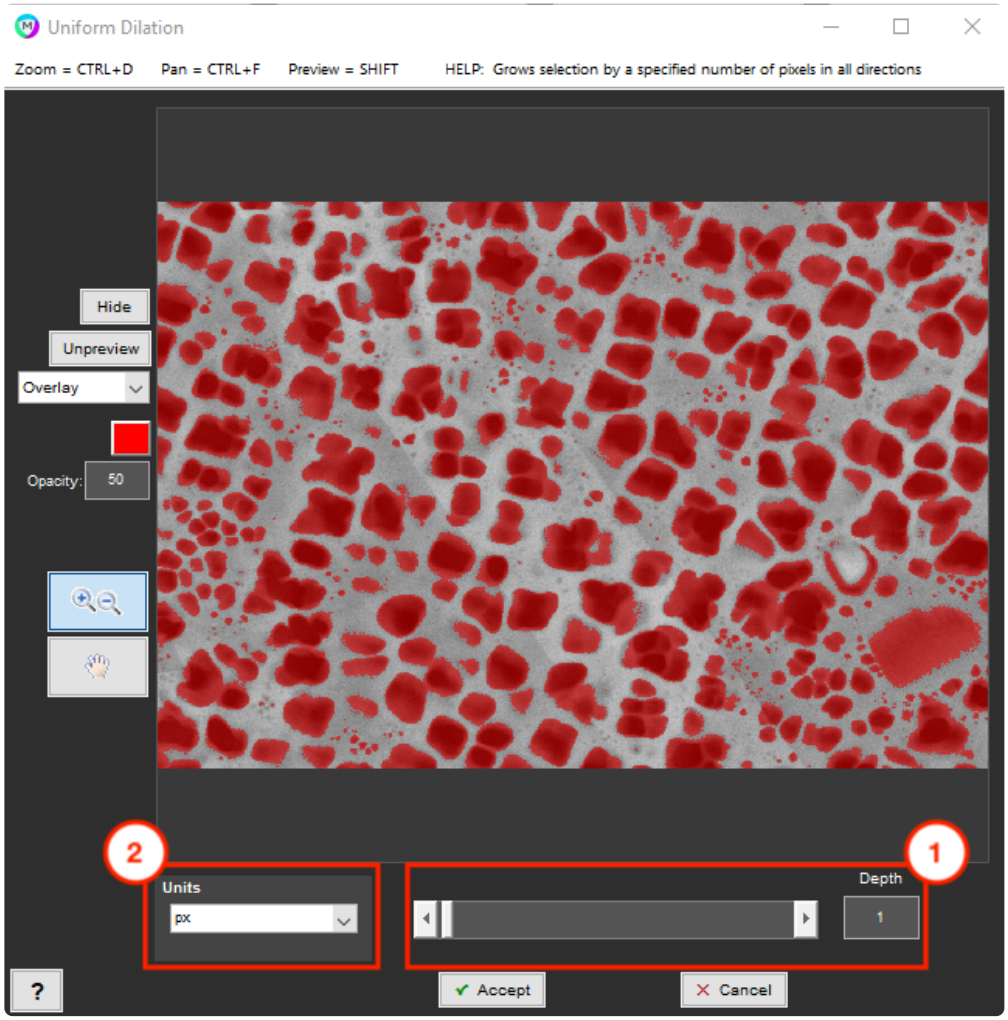
Functions

- Dilate Uniform
- Dilate Smart
- Dilate Retain

Dilate Uniform

Morphology > Dilate > Uniform

Grows selection by a specified number of pixels in all directions



1. Depth

The number of pixels by which selected regions will be grown in all directions

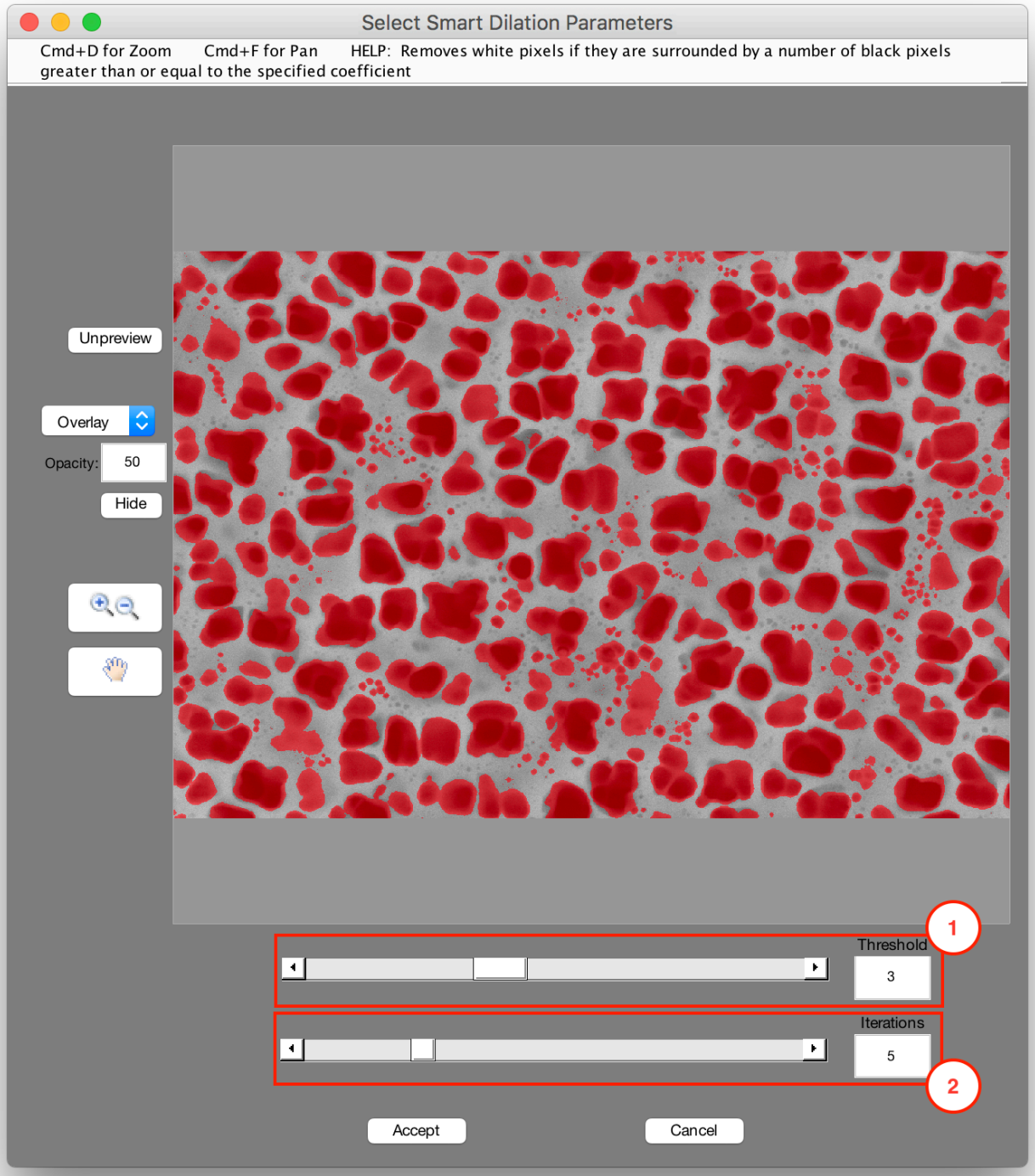
2. Units

Units for dilation depth. Choose between pixels and physical units (available when scale factor is set).

Dilate Smart

Morphology > Dilate > Smart

Selects pixels if they are surrounded by a number of selected pixels greater than or equal to the specified threshold number.



1. Threshold

The minimum number of selected pixels that must surround an unselected pixel for it to be added to the selection (Recommended: 5)

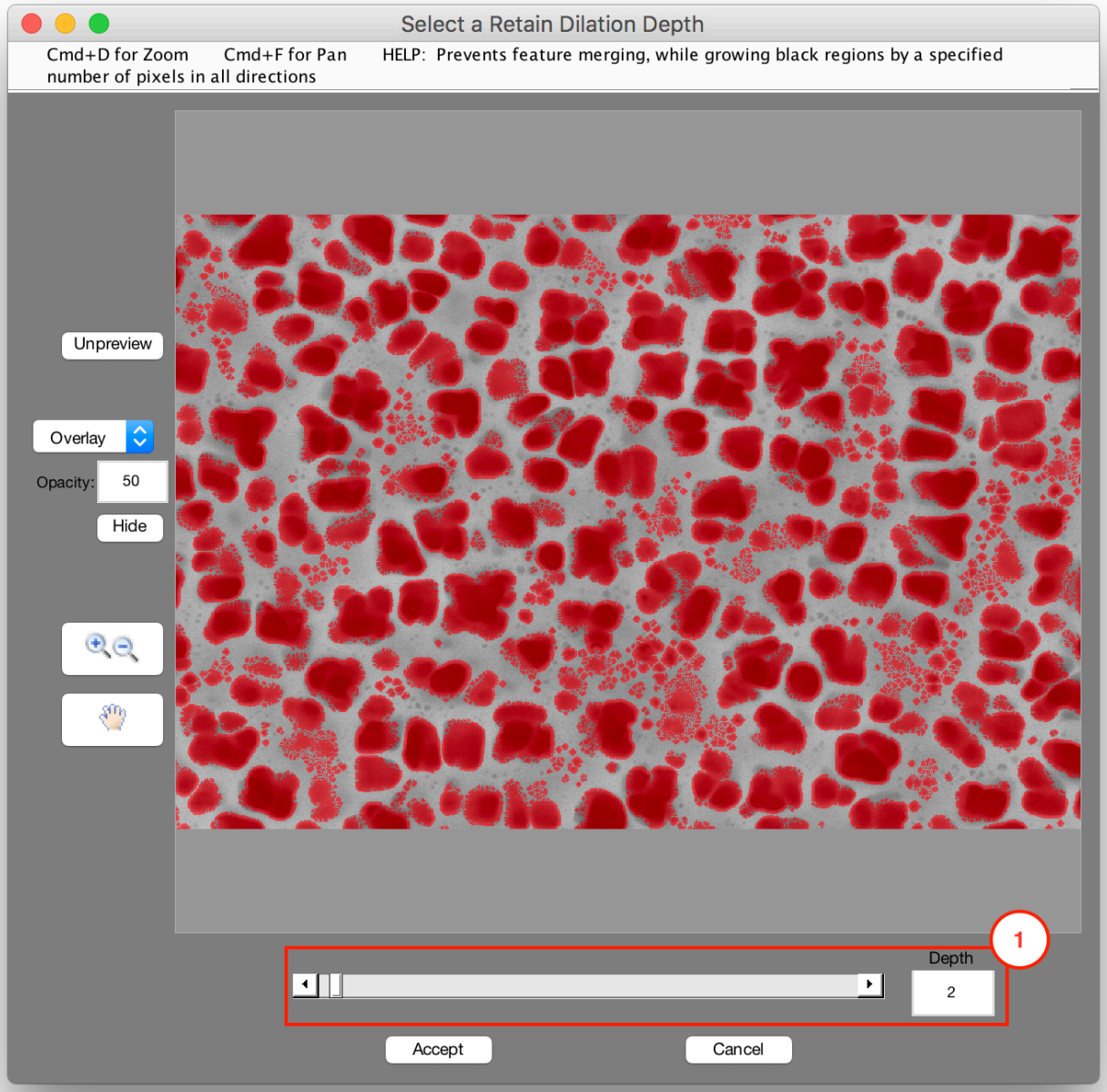
2. Iterations

The number of iterations to perform the dilation

Dilate Retain

Morphology > Dilate > Retain

Prevents feature merging, while growing selected regions by a specified number of pixels in all directions



1. Depth

The number of pixels by which selected regions will be grown in all directions without merging

Erode

Contains morphological operations which reduce a selection mask according to certain parameters.

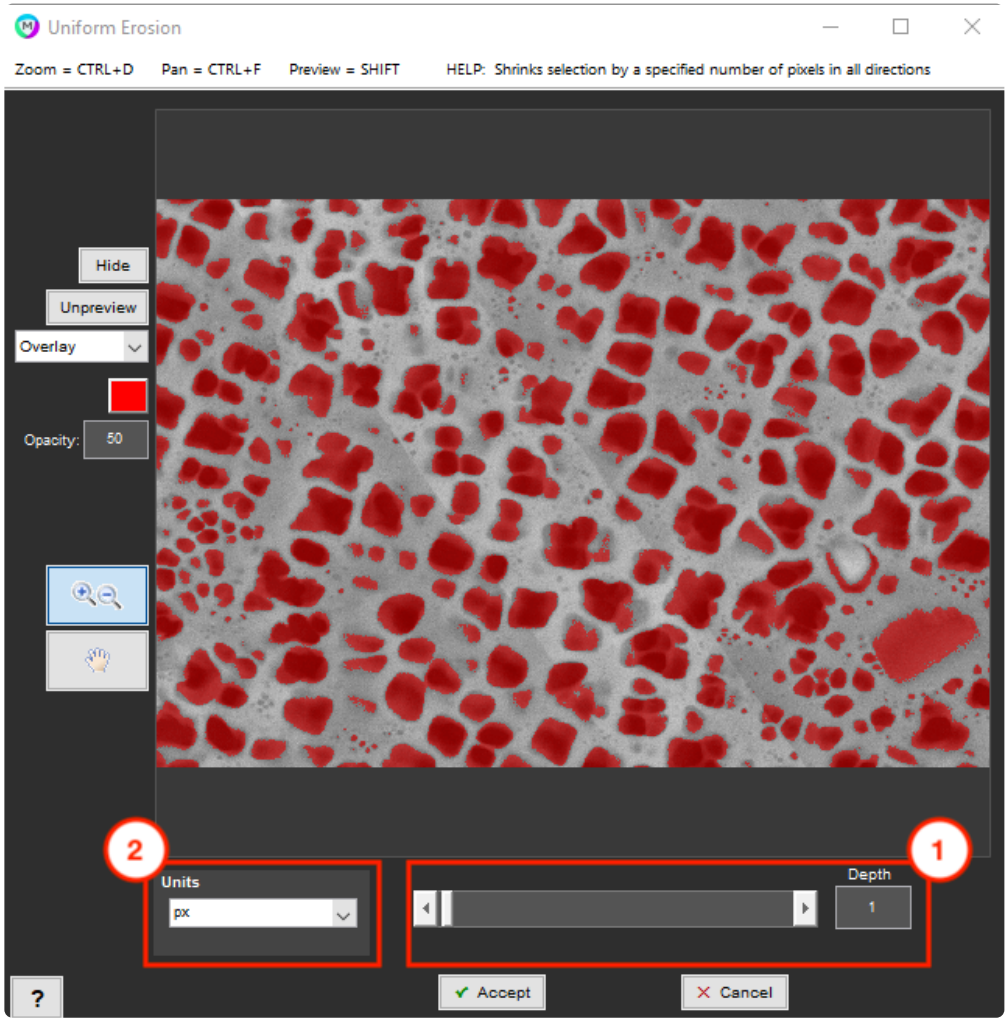
Functions

- Erode Uniform
- Erode Smart
- Erode Retain

Erode Uniform

Morphology > Erode > Uniform

Shrinks selection by a specified number of pixels in all directions



1. Depth

The number of pixels by which selected regions will be shrunk in all directions

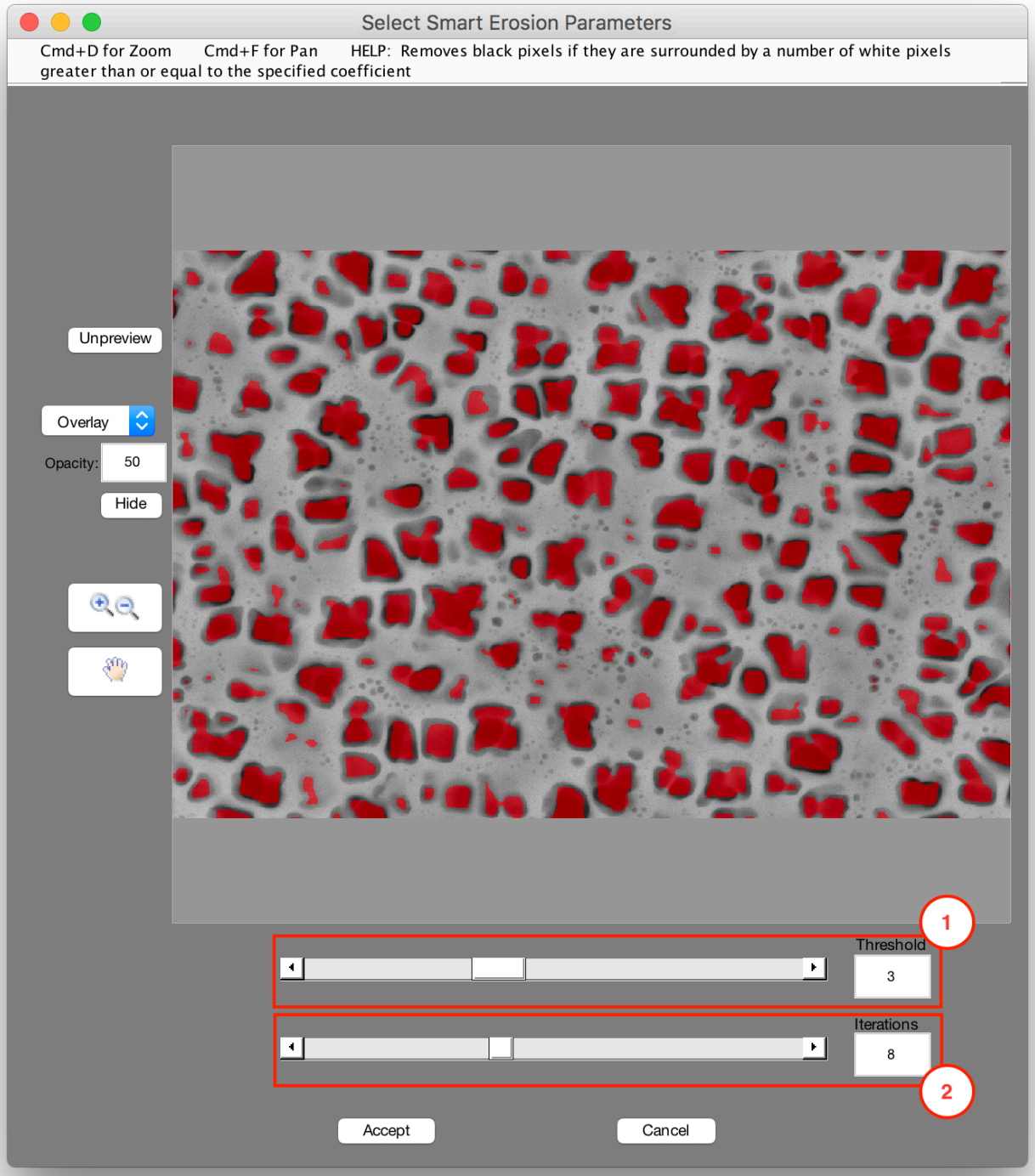
2. Units

Units for erosion depth. Choose between pixels and physical units (available when scale factor is set).

Erode Smart

Morphology > Erode > Smart

Removes selected pixels if they are surrounded by a number of empty pixels greater than or equal to the specified threshold number.



1. Threshold

The minimum number of empty pixels that must surround a selected pixel for it to be removed (Recommended: 5)

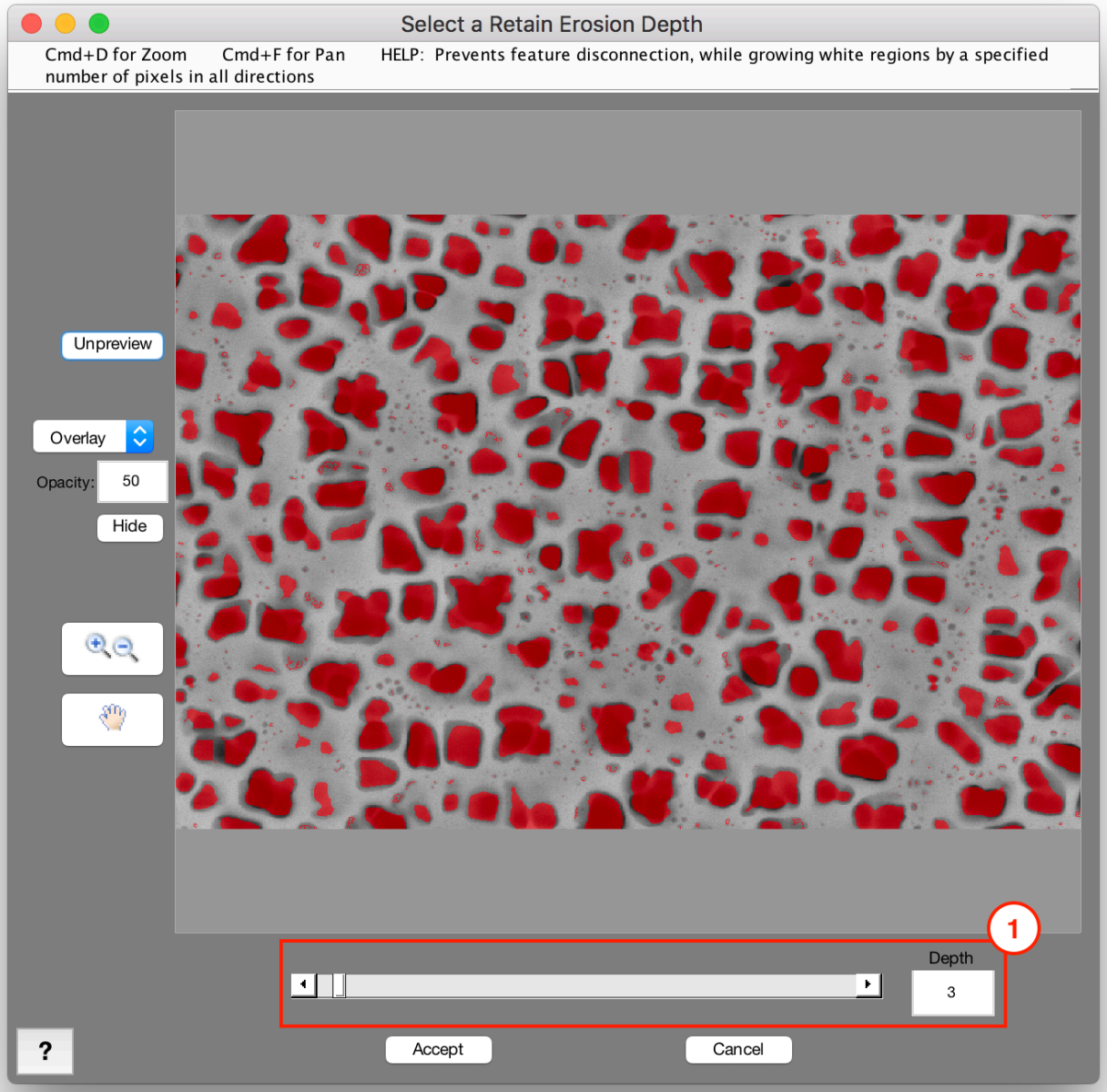
2. Iterations

The number of iterations to peform the erosion

Erode Retain

Morphology > Erode > Retain

Prevents feature disconnection, while shrinking selection by a specified number of pixels in all directions



1. Depth

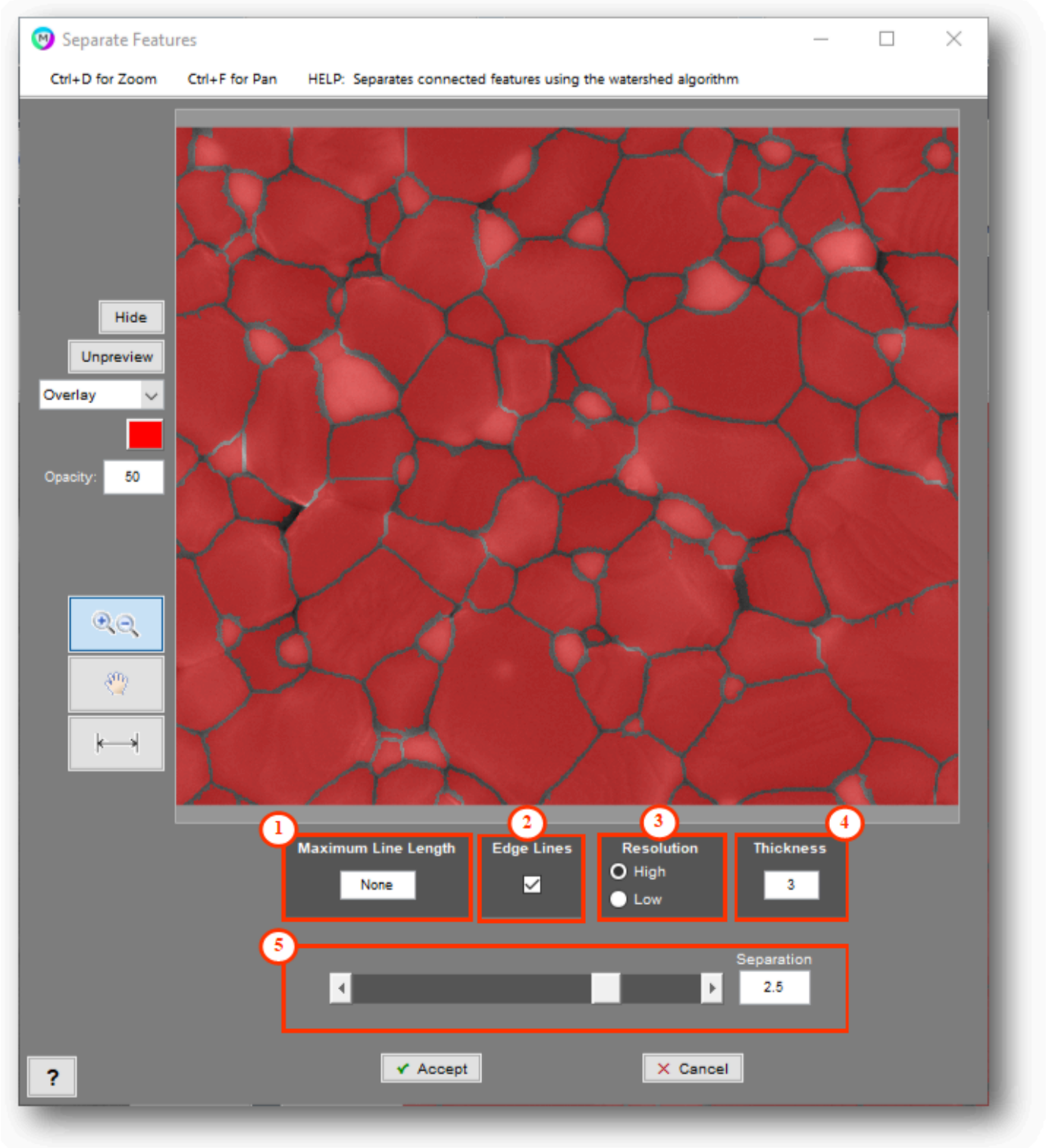
The number of pixels by which unselected regions will be grown in all directions without disconnecting

features

Separate Features

Morphology > Separate Features

Separates connected features using the watershed algorithm



1. Maximum Line Length

Set a maximum allowable line length for new separation lines.

2. Edge Lines

Toggle allowing separation lines to touch the edge of the image.

3. Resolution

- **High:** Works better for separating features with high pixel density
- **Low:** Works better for separating features with low pixel density (more pixelated)

4. Thickness

Thickness of separation lines

5. Separation

Controls strength of separation. Increase the slider to create more separations. (Recommended: 2-5)

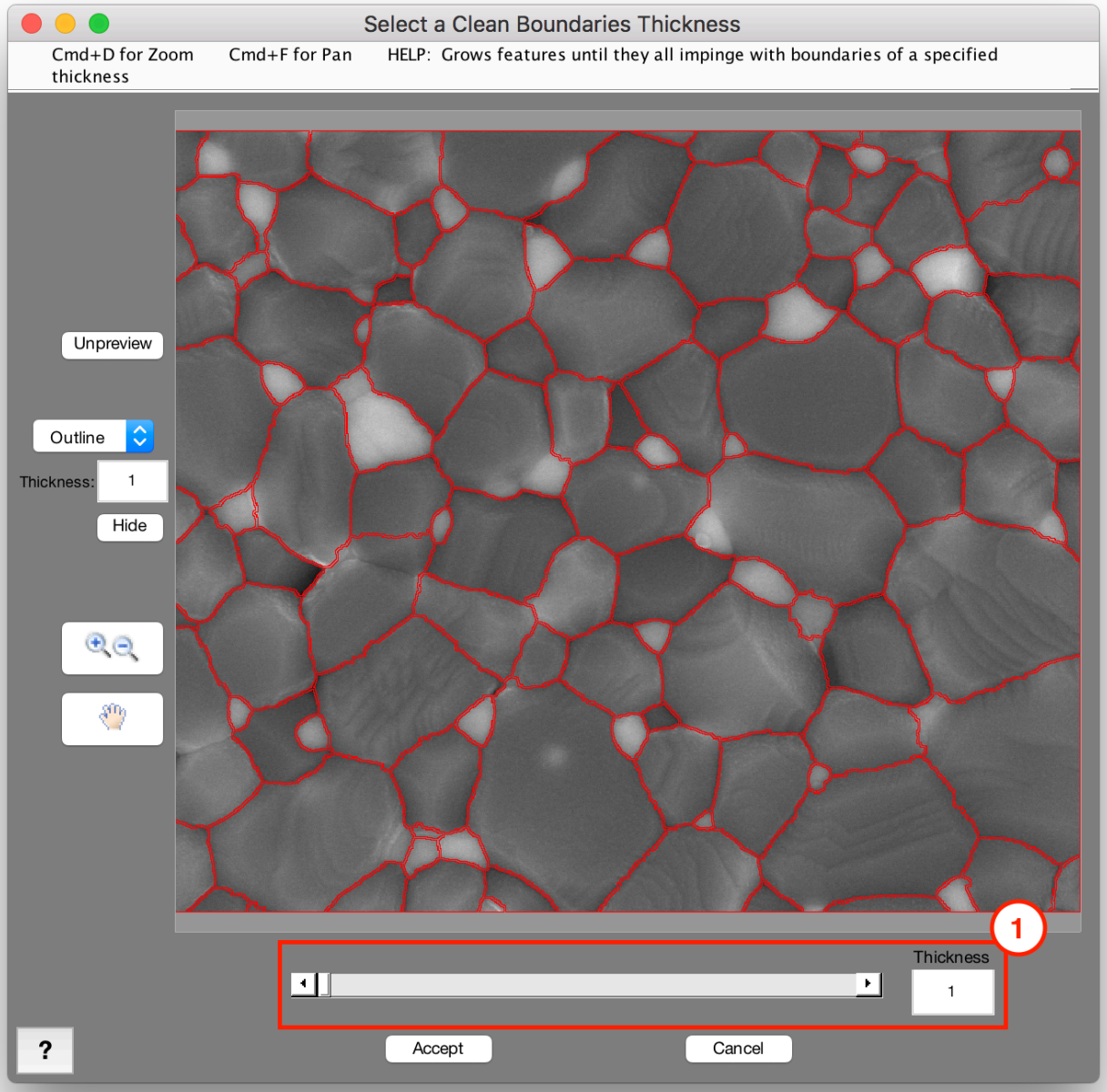
Tips

- Recommend starting separation value:3
- Match thickness to existing separation lines for best result
- In case of separation line artifact, isolate separation lines and reject lines based on size, orientation or roughness

Clean Boundaries

Morphology > Clean Boundaries

Grows features to fill the image until they are separated by boundaries of a specified thickness. This will also



1. Thickness

The resulting distance between features.

Notes

- Will not increase the border between features that are already closer than the specified thickness.
- Also completely fills any holes inside a feature.

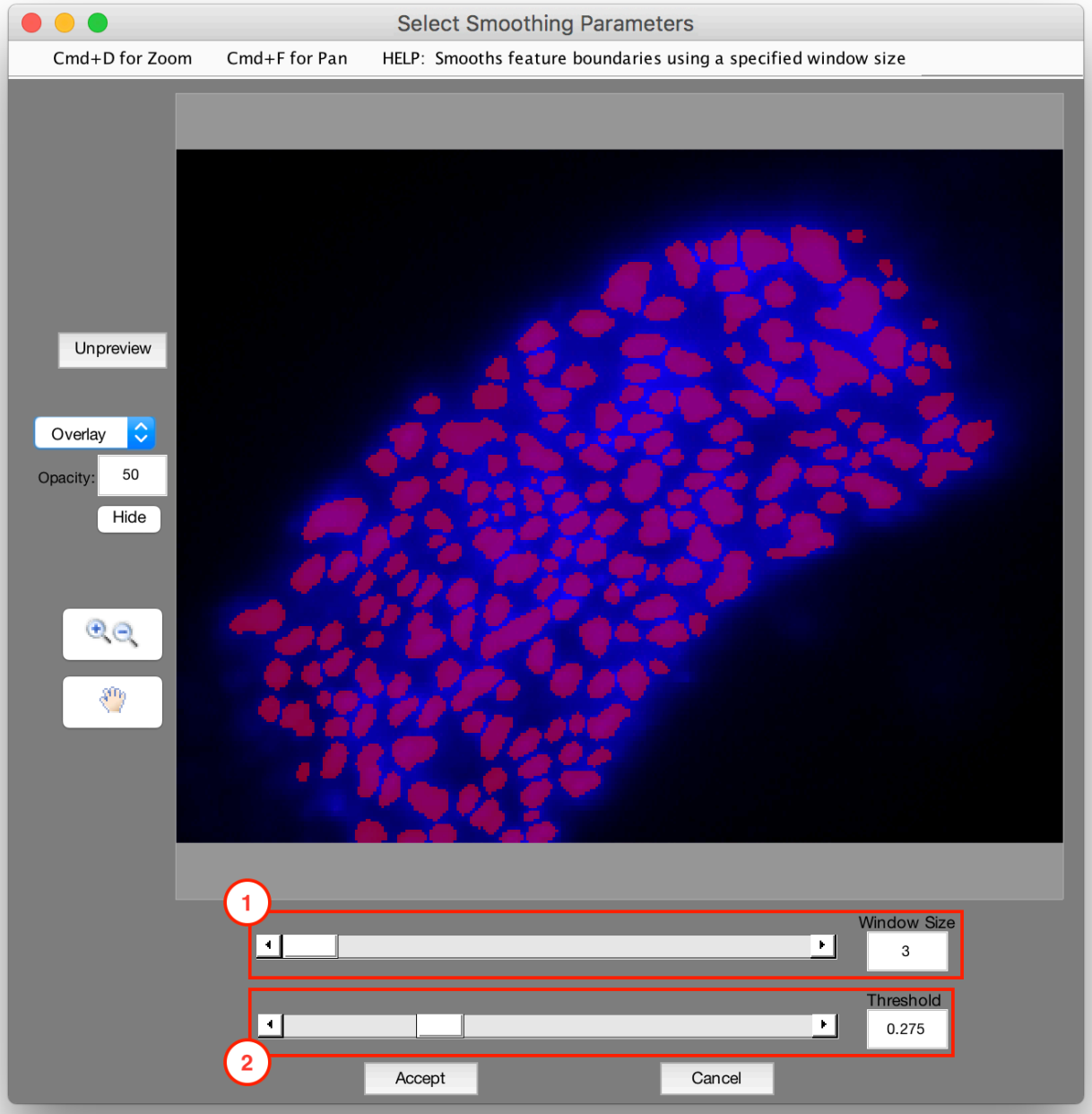
Tips

- Use step sequence ‘Clean Boundaries > Manual Edit > Clean Boundaries’ to quickly remove or add separation lines.

Smooth Features

Morphology > Smooth Features

Smooths feature boundaries using a specified window size. The threshold parameter controls the degree of feature erosion or dilation that also occurs.



1. Window Size

Window size which passes over each pixel for smoothing. Larger windows result in more smoothing.

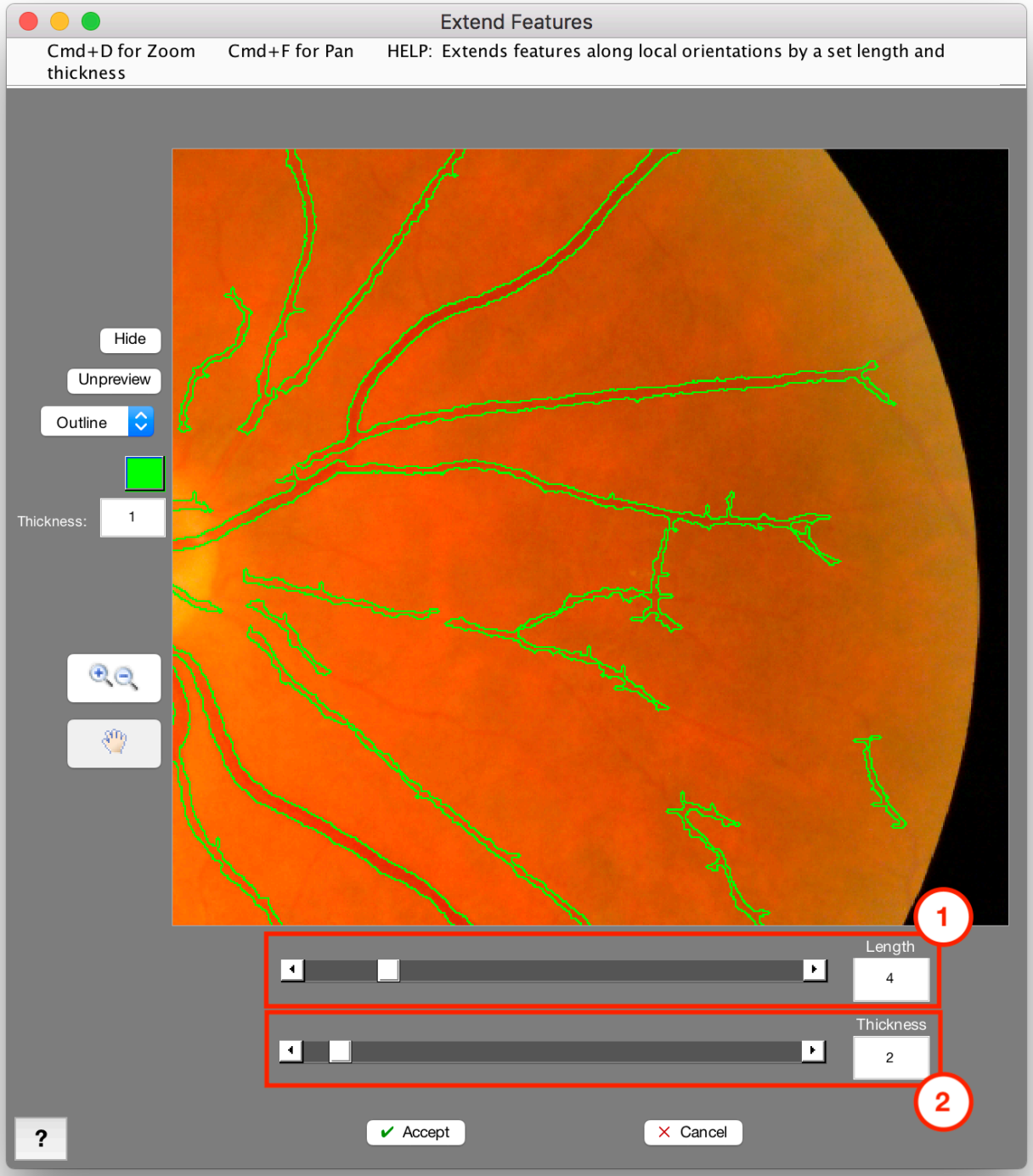
2. Threshold

Threshold value which controls erosion/dilation of features. Larger thresholds lead to less dilation/more erosion.

Extend Features

Morphology > Extend Features

Extends features along local orientations by a set length and thickness



1. Length

Length to extend features by (in pixels)

2. Thickness

Thickness of extended portions (in pixels)

Perimeter Pixels

Morphology > Perimeter Pixels

Sets pixels which are on the perimeter of features as empty, while all other pixels in an image are selected.

To select only the perimeter of your features, follow this step with an Invert.

Skeletonization

Morphology > Skeletonization

Skeletonize

Thins features down to their 1-pixel thick centerlines and branches without allowing features to break apart.

Keep Branch Points

Keeps skeleton branch points while removing the branches themselves. Works best on skeletons.

Remove Branch Points

Removes skeleton branch points to allow for subsets of branches to be removed (e.g., with Reject Features). Works best on skeletons.

Thin

Morphology > Thin

Thins features to skeletons with fewer branches than traditional skeletonization.

Pixel Lines

Morphology > Pixel Lines

Adds pixels to single-pixel lines so that each pixel in the line is connected on their faces. This ensures the lines are seen as fully connected features under the default “4-pixel” labeling connectivity.

Distance Map

Morphology > Distance Map

Generates a grayscale distance map of the current segmentation where each pixel is colored by its euclidean distance to the nearest empty pixel. White = zero distance, black = 255 distance.

Clean-Up

Contains functions for selectively removing features from the segmentation. Most commonly used is the Reject Features tool, which can remove or keep features based on a variety of shape and size metrics.

Functions

Reject

- Reject Features
- Fill All Holes
- Relative Size Filter
- Remove Edge Features

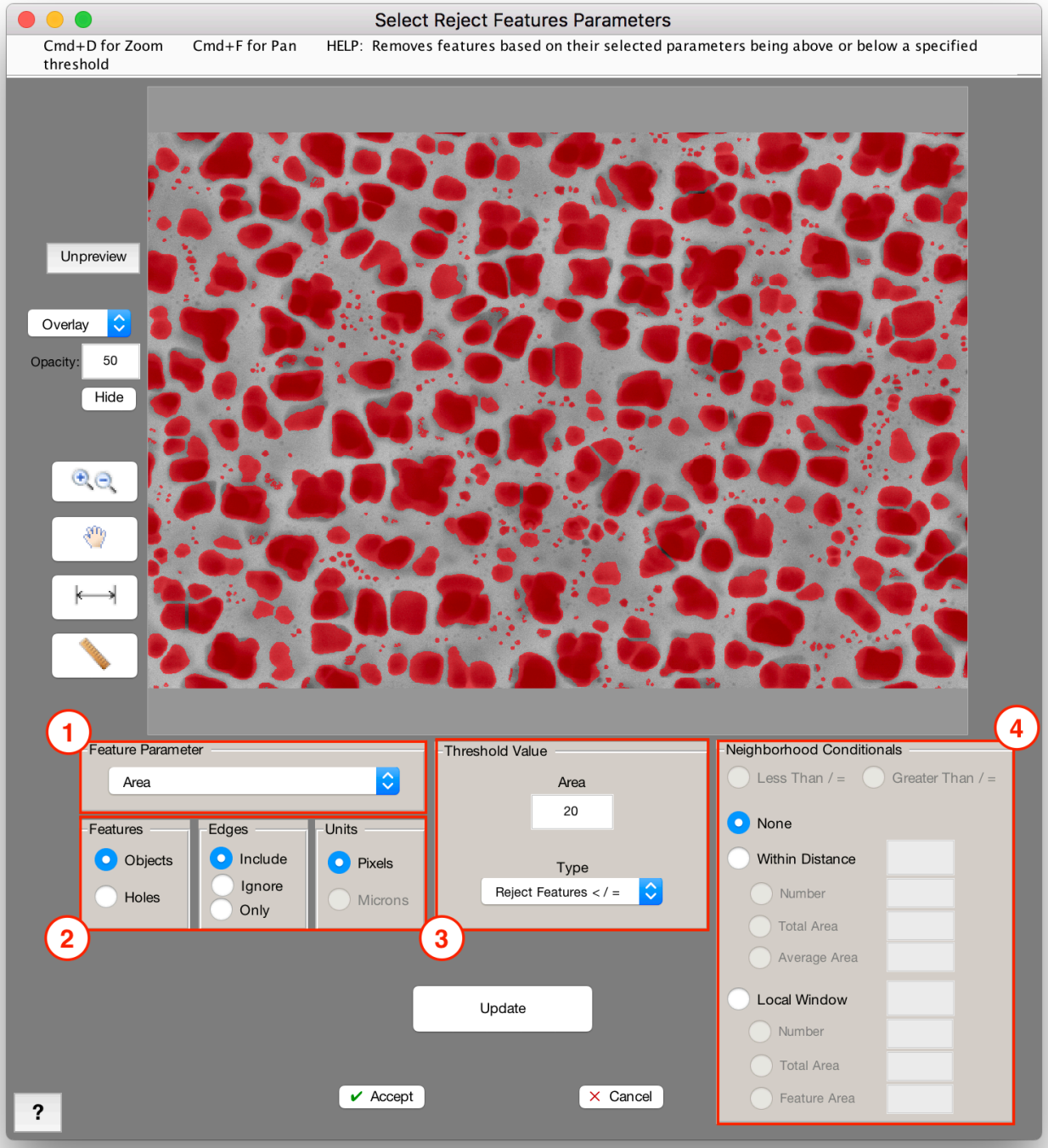
Replace

- Replace With
- Mark Center

Reject Features

Clean-up > Reject Features

Removes objects or fills holes based on their selected parameters being above or below a specified threshold. Parameter options include area, eccentricity, convex area/area ratio, and many more. Neighborhood conditionals are also optional, where features will only be removed if they meet both the parameter and neighborhood criteria.



1. Measurement

Determines the measurement that will be considered for each features potential rejection. Features will be rejected based on whether they are above or below the measurement threshold you specify.

- **Area:** Area of each feature.
- **Area Fraction:** Area fraction occupied by each feature relative to entire image.

- **Aspect Ratio**: Ratio of the major and minor axis lengths of each feature.
- **Average Neighbor**: Each feature's average distance to its neighbors, as defined by the features' Delaunay triangulation. Triangulation and distance are both calculated from the features' centroids.
- **Caliper Diameter**: Largest line length that fits across each feature. Equivalently, the distance between the two points in the feature farthest from each other, including those points.
- **Convex Area:** Area of each feature's tightest-fitting convex hull.
- **Eccentricity:** Describes how elongated or circular each feature is. 0 is a perfect circle. 1 is a straight line.
- **Equivalent Diameter:** Diameter of each feature if each was a circle of the same area.
- **Feature Number:** Numbers that are assigned to features when measurements are generated.
- **Filled Area:** Area of each feature with holes filled in.
- **Filled Area/Area:**
- **First Moment of Inertia:** First moment of each feature. Describes how much feature area is extended away from its centroid.
- **First Moment of Inertia/Area:** Ratio between the first moment of each feature and its area.
- **Intensity Mean:** *(requires Companion Image)* Grayscale intensity average of the entire Current Image.
- **Intensity StdDev:** *(requires Companion Image)* Grayscale intensity standard deviation of the entire Current Image.
- **Intensity Sum:** *(requires grayscale Companion Image)* Grayscale intensity sum over the entire Current Image.
- **Length – X:** Length of each feature's bounding box in the X-direction.
- **Length – Y**: Length of each feature's bounding box in the Y-direction.
- **Major Axis Length:** Major axis length of ellipse fit to each feature.
- **Minor Axis Length:** Minor axis length of ellipse fit to each feature.
- **Moment Invariant (Omega-1):** High-order moment which describes shape properties of each feature [1,2].
- **Moment Invariant (Omega-2):** High-order moment which describes shape properties of each feature [1,2].
- **Moment Invariant (Phi-1):** High-order moment which describes shape properties of each feature [1,2].
- **Moment Invariant (Phi-2):** High-order moment which describes shape properties of each feature [1,2].
- **Nearest Neighbor**: Each feature's distance to the closest other feature, calculated from the features' centroids.
- **Number of Features:** *(requires Companion Image)* Number of features in the companion image contained within the feature.
- **Number of Holes:** Number of holes contained within the feature.
- **Orientation:** Angle of the ellipse fit to each feature with respect to the positive X-axis. Positive angles are clockwise rotations and negative counterclockwise.
- **Perimeter:** Length of perimeter of each feature.
- **Perimeter/Area:** Perimeter of each feature relative to its area.
- **Roughness:** Ratio between the area of tightest-fitting convex hull and the area of each feature.
- **Roundness:** Ratio of equivalent diameter to caliper diameter (see above).

2. Features, Edge Features, Units

- **Features**: You can choose which features you want to be rejected between white and black.
- **Edge Features**: You can choose to include or ignore edge features when choosing features to reject. You can also choose to only reject edge features.
- **Units**: The units you would like use to reject features.

3. Threshold Value

Select the threshold value (of the set measurement) for rejecting features and specify whether you would like to reject or keep features below the threshold value.

4. Proximity Awareness

Lets you consider the features above or below the threshold as *candidates* for removal, but only actually removed them if their neighbors pass the specified threshold.

- **Less Than / =, Greater Than / =:** Choose whether neighboring features must be less than or equal to, or greater than or equal to, in order to pass the threshold.
- **None:** Do not use neighborhood conditionals
- **Within Distance**: Consider features whose centroids are within certain distance. Enter distance in which to search (in pixels);
  - **Number:** Consider number of features. Enter critical number of features.
  - **Total Area:** Consider total area of features. Enter critical total area (in pixels).
  - **Average Area:** Consider average area of features. Enter critical average area (pixels).
- **Local Window**: Consider local window about each feature, including partial neighboring features.
  - **Number:** Consider number of features. Enter critical number of features.
  - **Total Area:** Consider total area of features. Enter critical total area (in pixels).
  - **Average Area:** Consider average area of features. Enter critical average area (pixels).

References

[1] Rosin, Paul L. 2003. “Measuring Shape: Ellipticity, Rectangularity, and Triangularity.” Machine Vision and Applications 14 (3): 172–184.

[2] MacSleyne, J P, J P Simmons, and M De Graef. 2008. “On the Use of Moment Invariants for the Automated Analysis of 3D Particle Shapes.” Modelling and Simulation in Materials Science and Engineering 16 (4) (June 1): 045008.

Fill All Holes

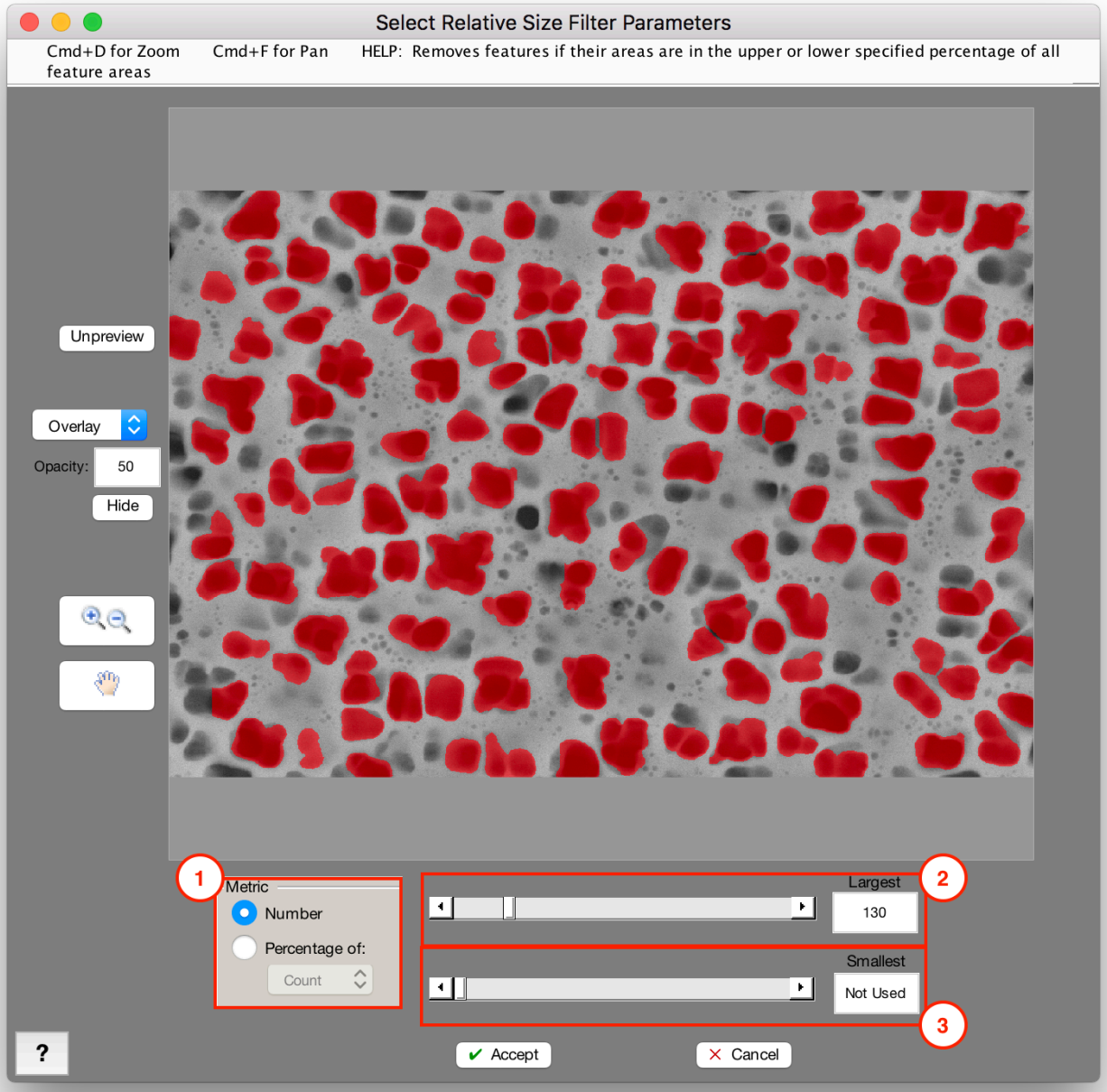
Clean-Up > Fill All Holes

Removes all holes contained within features. If this unintentionally removes large areas of trapped empty space, try using Reject Features to only remove holes below a certain area.

Relative Size Filter

Clean-Up > Relative Size Filter

Removes features if their areas are in the upper or lower specified percentage of all feature areas.



1. Metric

- **Number:** Select largest/smallest number of features

- **Percentage:** Select largest/smallest percentage of features

2. Largest

Threshold number or percentage to select the largest features

3. Smallest

Threshold number or percentage to select the smallest features

Tips

- Avoid using strictly number threshold unless confident in the number of features that should be removed or kept.

Remove Edge Features

Clean-Up > Remove Edge Features

Removes features if they contact the edge of the image.

Tips

- Useful for removing partial features in the image; recommended for generating representative size measurements.

Replace With

Clean-Up > Replace With

Replace each feature in the Current Image with the selected representative feature.

Box

Replaces each feature with its bounding box.

Caliper Diameter

Replaces each feature with its caliper diameter.

Centroid

Replaces each feature with its centroid.

Convex Hull

Replaces each feature with its convex hull.

Ellipse

Replaces each feature with its equivalent ellipse.

Equivalent Circle

Replaces each feature with its equivalent circle.

Fit Circle

Replaces each feature with its best fitting circle.

Largest Circle

Replaces each feature with its largest circle. The largest circle is defined as the circle whose diameter is the feature’s caliper diameter, and whose center is the midpoint of the line segment drawn along the caliper diameter.

Local Normal

Replaces each feature with its local normal based on surrounding features.

Major Axis

Replaces each feature with the major axis of its equivalent ellipse.

Minimum Diameter

Replaces each feature with its minimum diameter.

Minor Axis

Replaces each feature with the minor axis of its equivalent ellipse.

Nearest Distance

Replaces each feature with a line between it and its nearest neighbor.

Triangulation

Replaces all features with a Delaunay triangulation calculated from the features' centroids.

Mark Center

Clean-Up > Mark Center

Mark the center of a blank image with a selected pixel

Memory

Contains functions for storing and recalling images from various image memory slots.

Functions

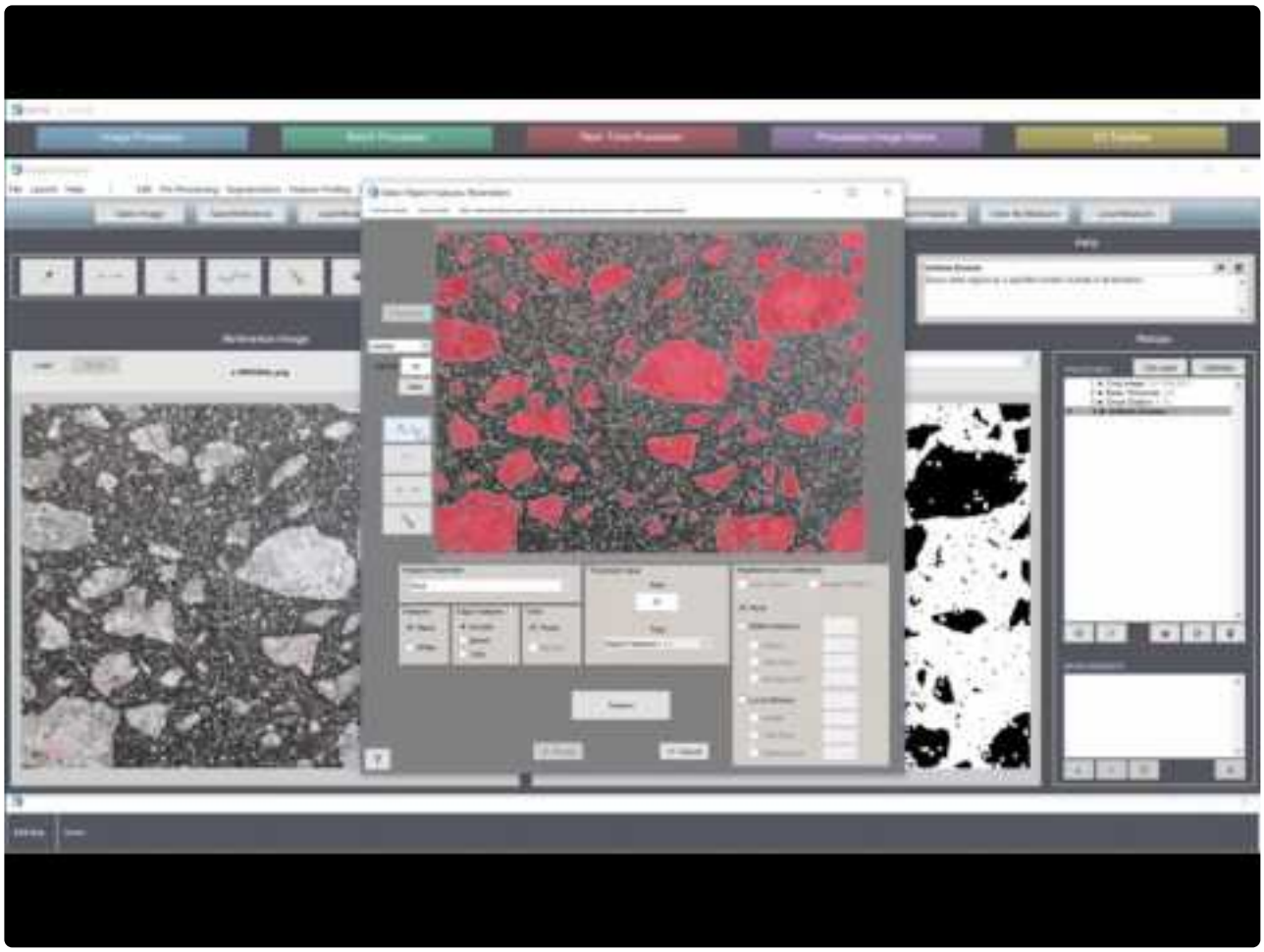
Companion Image

- Set Companion Image
- Load Companion Image
- Call Companion Image

Memory

- Set Memory Image #1-6
- Call Memory Image #1-6
- Call Original Image

Tutorial



https://www.youtube.com/embed/ex7UHo7tvro?rel=0

https://www.youtube.com/embed/ex7UHo7tvro?rel=0

Set Companion Image

Memory > Set Companion Image

Stores the Current Image in memory as the Companion Image. This is the most important memory slot as it is used for various arithmetic, masking, and measurement operations between it and the Current Image.

The companion image will maintain the original bit depth when the following steps are immediately before the Set Companion step:

- Call Original Image
- Load Companion Image
- Channel Operation

Call Companion Image

Memory > Call Companion Image

Call the Companion Image to be the Current Image.

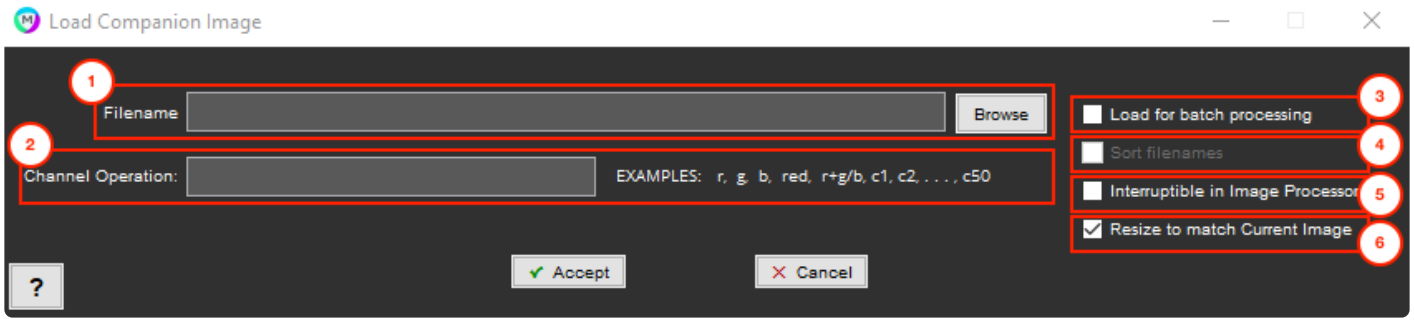
Load Companion Image

Memory > Load Companion Image

Loads in an image from a file as the Companion Image and as the Current Image.

If the chosen image's directory will hold a set of images to be paired with starting images in a batch process, then select check the "Load for batch processing" checkbox. Otherwise, the single chosen image will be used in each batched application of the recipe.

The Companion Image is the most important memory slot as it is used for various arithmetic, masking, and measurement operations between it and the Current Image.



1. Filename

File to load as Companion Image

2. Channel Operation

Enter Channel Operation. Only applicable to color images. See here for more information.

3. Load for Batch Processing

Successively load files from "Load Companion" directory during a batch process.

4. Sort Filenames

Sort files in "Load Companion" directory prior to successive loading.

5. Interruptible in Image Processor

Pauses recipe execution on load in Image Processor and prompts user with this window.

6. Resize to match Current Image

Forces Companion Image to resize to match that of the Current Image, in the event they are different.

Set Memory Image #1-6

Memory > Set Memory Image #1-6

Stores the Current Image in memory as Memory Image #1-6. You can store up to six different images. These images can be recalled later as well as used in some masking operations.

Call Memory Image #1-6

Memory > Call Memory Image #1-6

Calls Memory Image to be the Current Image. You can call up to six different images to be the Current Image.

Call Original Image

Memory > Call Original Image

Calls the Original Image to be the Current Image.

Math

Contains functions for performing mathematical operations between two images. These operations can range from blending two grayscale images together, to only keeping features in one image that contact features in another.

Functions

Operations on selection masks (a.k.a. black-and-white images)

- *Feature Union
- *Feature Minus
- *Feature Intersection
- *Keep Mutual Features
- *Keep Exclusive Features
- Make Grayscale

Operations on grayscale images

- *Add
- *Average
- *Divide
- *Multiply
- *Subtract
- Add Value…

Operations on both black-and-white and grayscale images

- *Merge Darker Pixels
- *Merge Lighter Pixels

*Union

Math > *Union

Requires Companion Image

Combines the features between the Current and Companion Images. All features from both images will appear in the result.

*Minus

Math > *Minus

Requires Companion Image

Subtracts the Companion Image features from the Current Image features.

*Intersection

Math > *Intersection

Requires Companion Image

Keeps the areas of features which overlay between the Current and Companion Images. Only overlapping feature areas from both images will appear in the result.

*Keep Mutual

Math > *Keep Mutual

Requires Companion Image

Keep features from the Current Image that contact or overlap a feature in the Companion Image.

*Keep Exclusive

Math > *Keep Exclusive

Requires Companion Image

Keep features from the Current Image that do not contact or overlap a feature in the Companion Image.

Make Grayscale

Math > Make Grayscale

Convert a binary Current Image into a grayscale image. Selected pixels become 0, while deselected pixels become 255.

*Add

Math > *Add

Requires Companion Image

Adds the Current Image to the Companion Image.

*Average

Math > *Average

Requires Companion Image

Averages the Current Image with the Companion Image.

*Divide

Math > *Divide

Requires Companion Image

Divides the Companion Image into the Current Image.

*Multiply

Math > *Multiply

Requires Companion Image

Multiplies the Companion Image into the Current Image.

*Subtract

Math > *Subtract

Requires Companion Image

Subtracts the Companion Image from the Current Image.

Add Value

Math > Add Value

Add a constant integer value to each pixel in a grayscale Current Image. May be positive or negative. Resultant pixel values below 0 or above 255 are clipped to 0 or 255.

*Merge Darker Pixels

Math > *Merge Darker Pixels

Requires Companion Image

Merges the darker pixels between the Current and Companion Images. Each pixel in the resulting will be the darkest value of that pixel in the Current and Companion images.

When merging a grayscale image with a binary selection, selected = black (0) and unselected = white (255).

*Merge Lighter Pixels

Math > *Merge Lighter Pixels

Requires Companion Image

Merges the lighter pixels between the Current and Companion Images. Each pixel in the resulting will be the lightest value of that pixel in the Current and Companion images.

When merging a grayscale image with a binary selection, selected = black (0) and unselected = white (255).

View

Contains functions for viewing the Current Image's histogram, finding a specific feature by its ID number, and managing various properties of Recipe Layers.

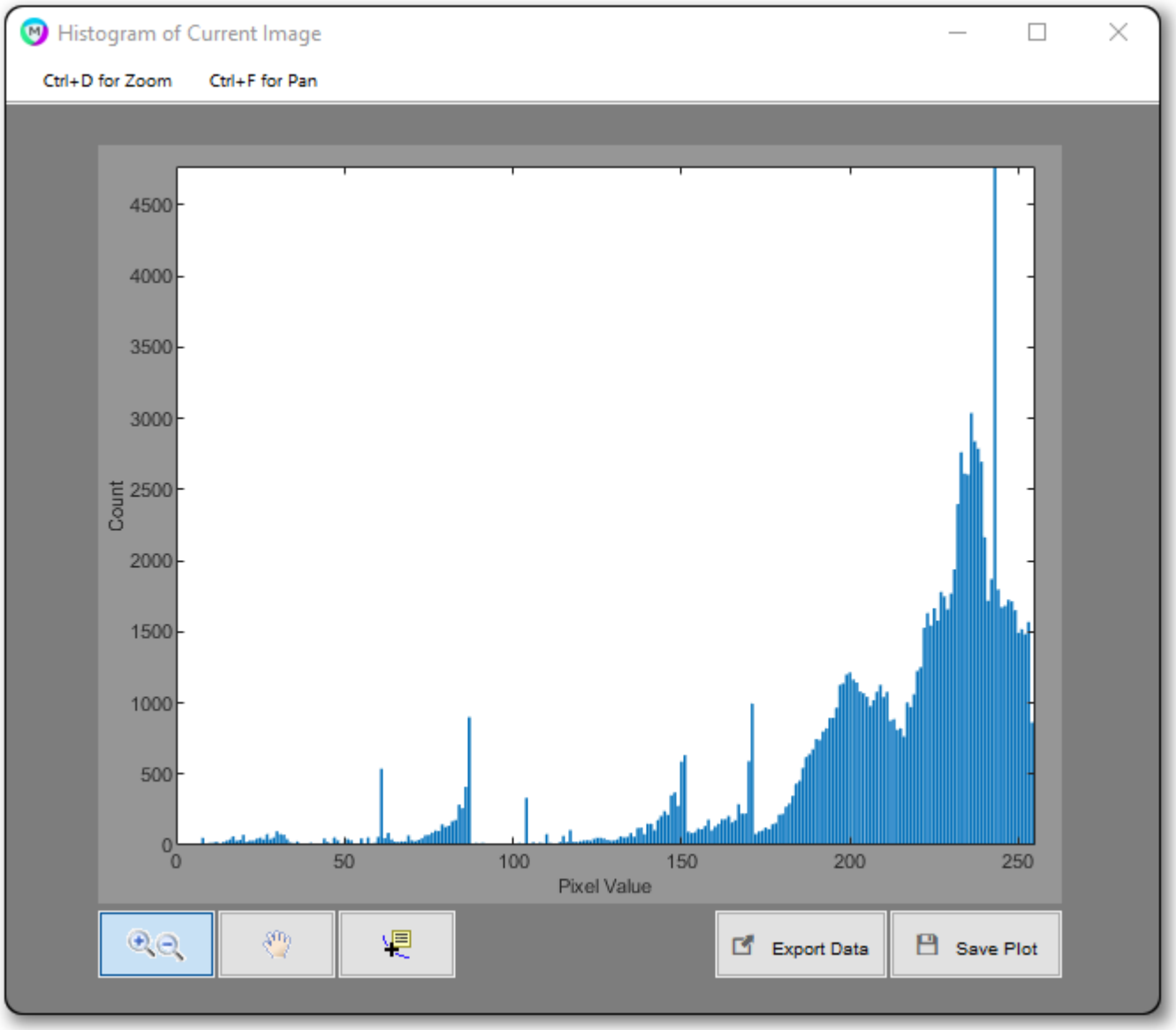
Functions

- Image Histogram
- Specific Feature
- Manage Layers

Image Histogram

View > Image Histogram

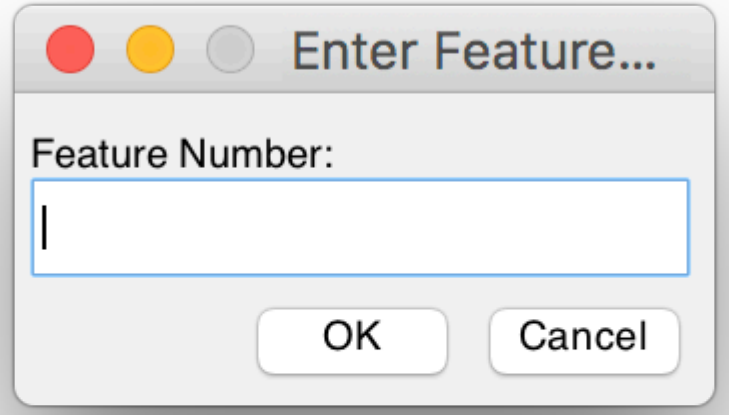
Histogram of the current image. For binary selections, selected = 0 and unselected = 1.



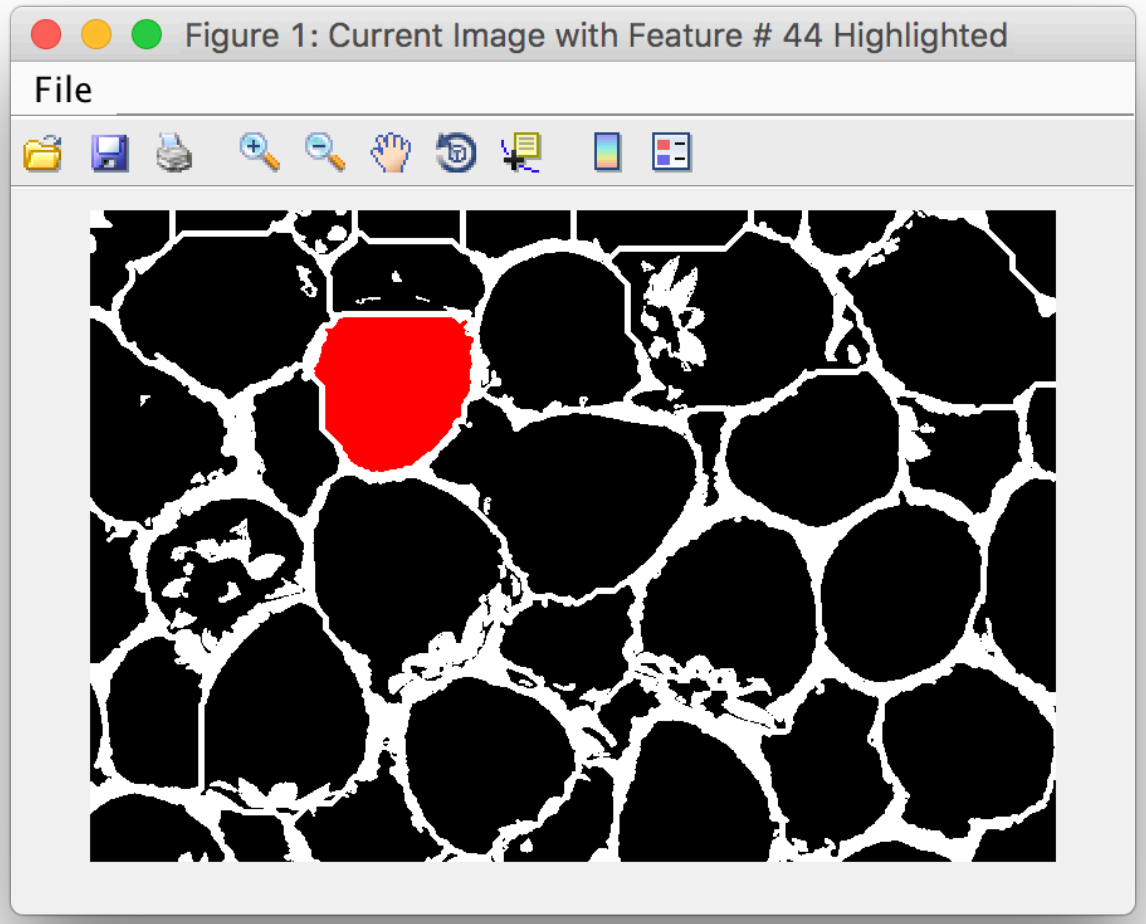
Specific Feature

View > Specific Feature

Specific Feature allows you to view where the feature is in the image by highlighting.



The list of feature numbers can be found by generating feature measurements. Then, you will be able to tell what the maximum number of features you have.



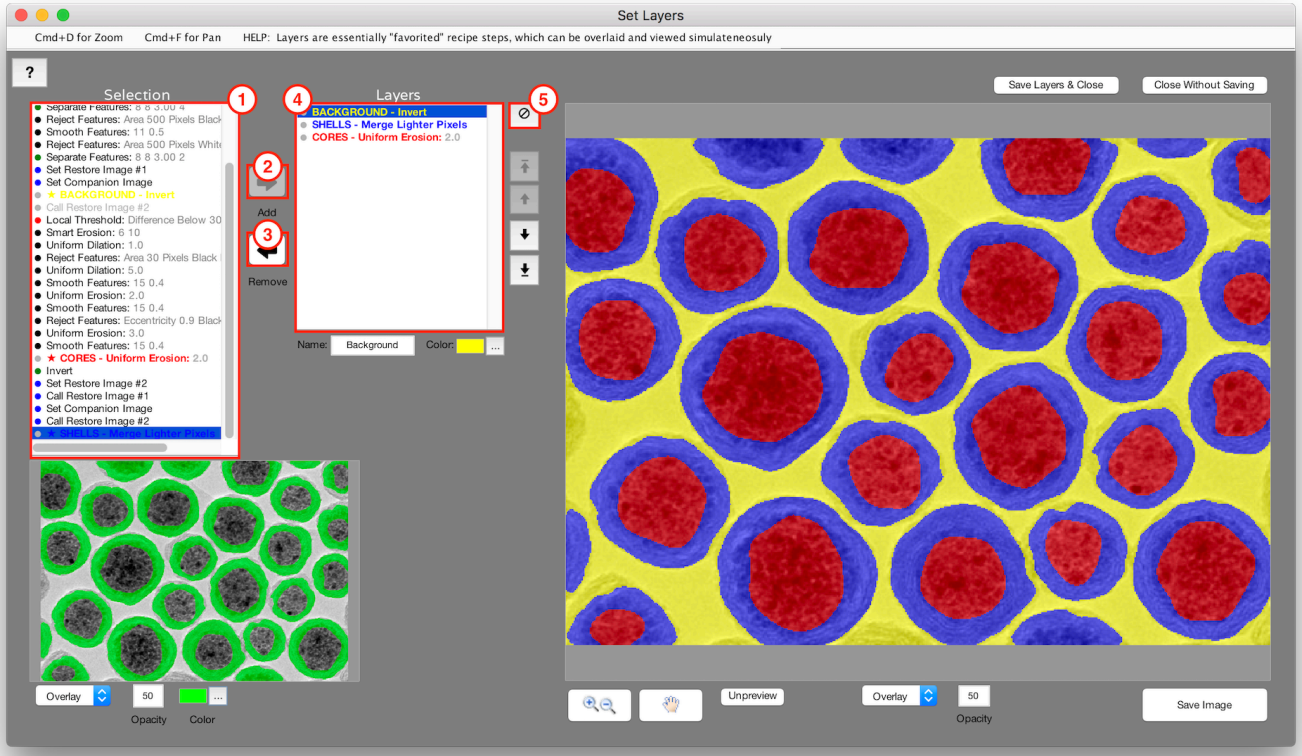
The image can be saved or printed. The feature number you entered will be highlighted in the image.

Manage Layers

View > Manage Layers

While Layers are most conveniently set, edited, and removed from the controls directly in the Image Processor, this window allows more control over Layer positioning. It also provides a comprehensive management center for your Layers.

Layers are essentially “favorited” recipe steps, which can be overlaid and viewed simultaneously. Any measurements in the recipe will be performed on layers when run in a batch and the result of any “layered” step will be saved and, if applicable, added as a layer to a 3D reconstruction. If there are no layers in a recipe, then any batch measurements, image saves, and reconstruction layers will just come from the last step in the recipe.



1. Selection

Click on a Recipe step to view its result in the viewer below this panel. B/W Recipe steps can be set as Layers using the “Add Layer” button.

2. Add Layer

Added selected Recipe step as a Layer

3. Remove Layer

Remove selected Layer

4. Layers

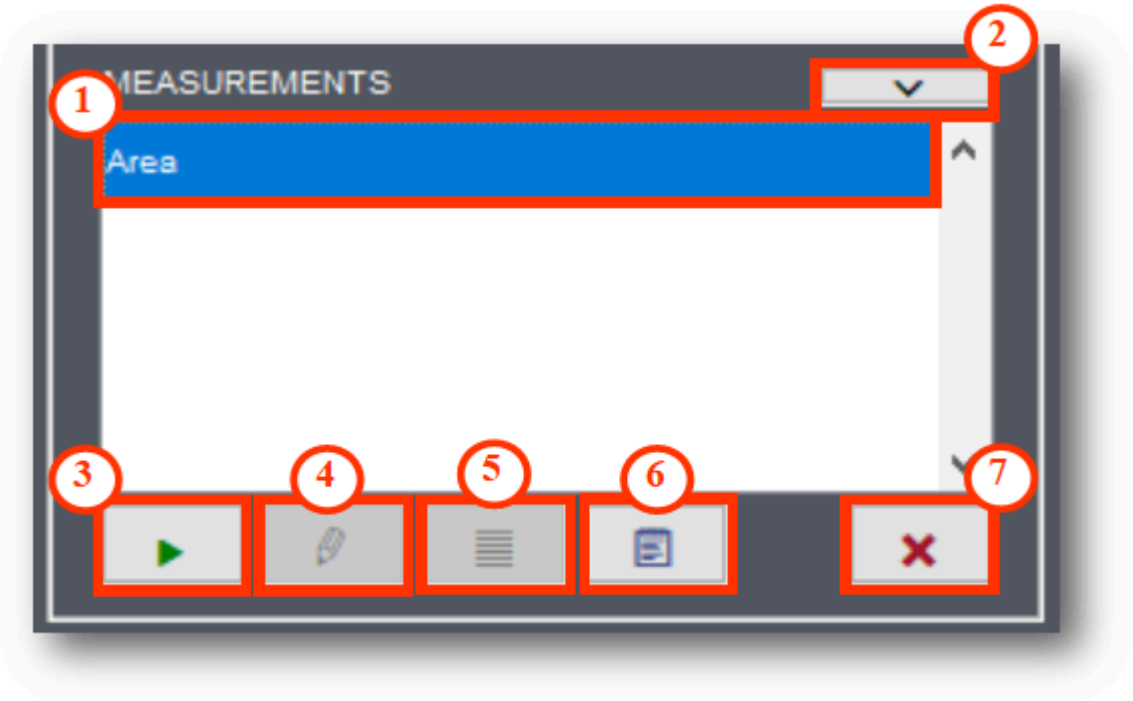
Currently set Layers. Layer names, colors, and overlay position can be set from the surrounding controls.

5. Show/Hide Layer

Show/hide selected Layer in the viewer on the right

Measurements

Layout



1. Active measurement

2. Show/Hide Measurements Panel

3. Run active measurement on active Recipe step or shown Layers

4. Edit active measurement settings

5. Edit which Layers the active measurement is executed on

6. Generate measurements report for all Layers and measurements

7. Delete measurement

Functions

Per Image

- Area

- Area Fraction
- Count
- Estimate Count
- Intercepts
- Image Dimensions
- Number Density
- Perimeter
- Perimeter Fraction
- *Intensity Mean
- *Intensity StdDev
- *Intensity Sum
- *Correlation Coefficient
- *Mutual Information

  (* Starred measurements require a Companion Image)

Per Feature

- Feature Measurements
- Color by Measurements

Per Pixel

- Local Measurements

Area

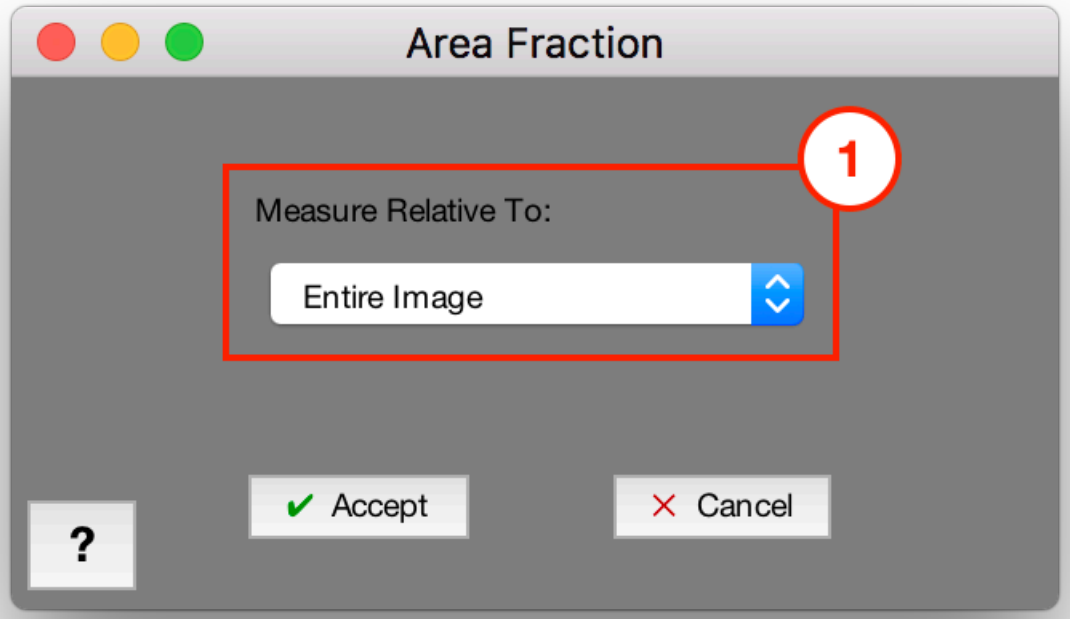
Measurements > Area

Measures the total area of selected pixels in the Current Image.

Area Fraction

Measurements > Area Fraction

Measures the area fraction of selected pixels in the Current Image. Counts all selected pixels and divides the number by either the total number of pixels in the image, or the total number of selected pixels in a chosen Memory Image.



1. Measure Relative To

Choose to measure relative to either the entire image, or the selection in a Memory Image

Count

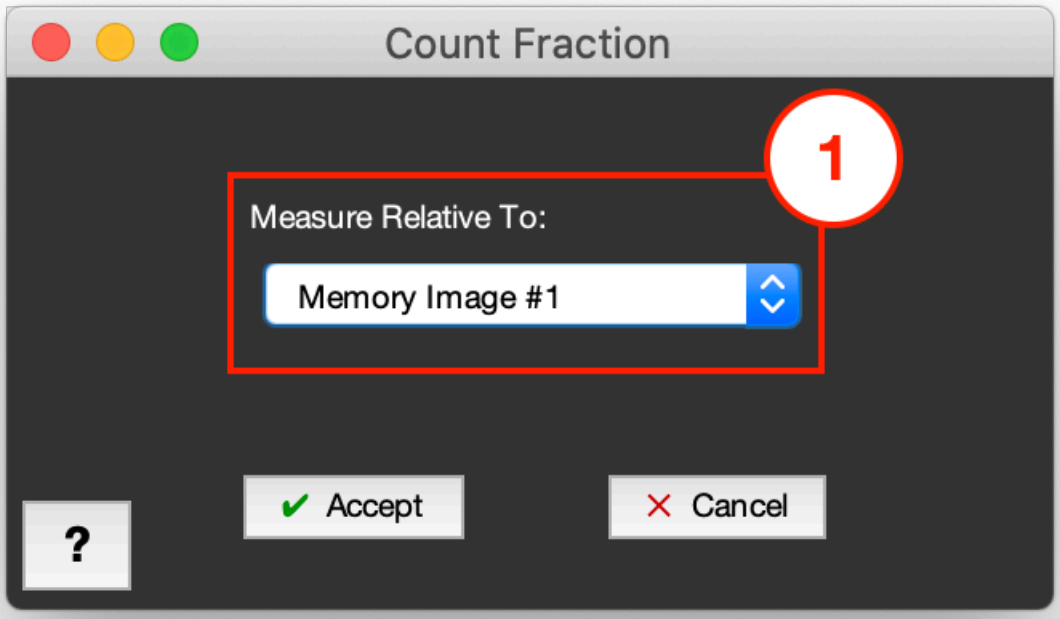
Measurements > Count

Measures the total number of separate features in the Current Image.

Count Fraction

Measurements > Count Fraction

Measures the count of features in the Current Image as a percentage relative to the count of features in a chosen Memory Image.



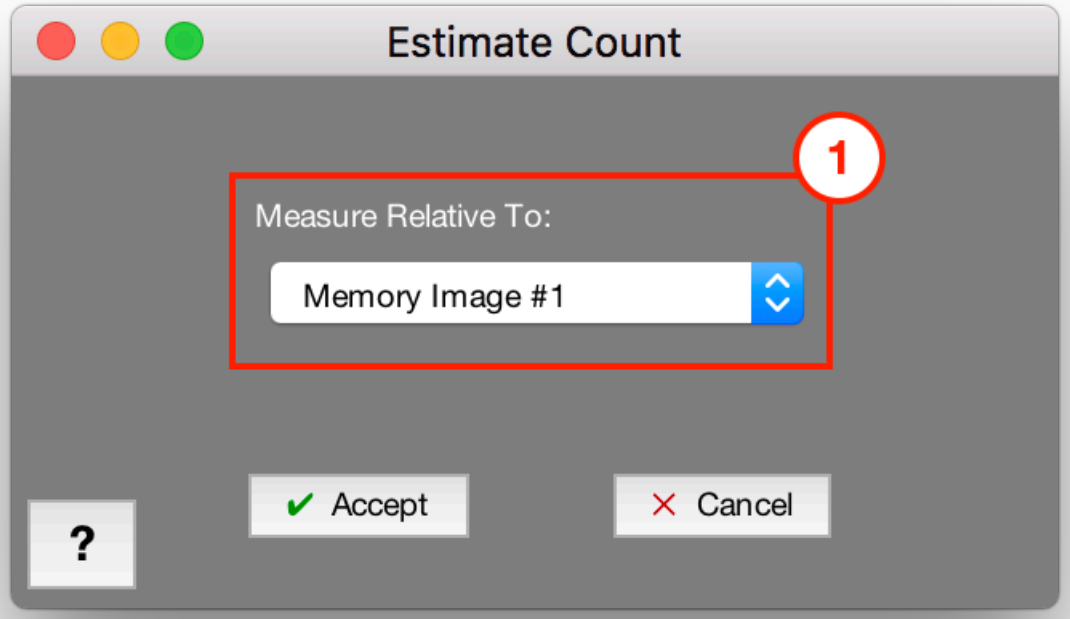
1. Measure Relative To

Choose which Memory Image to use to measure the count fraction. The feature count in the Current Image is divided by the feature count in the chosen Memory Image and expressed as a percentage.

Estimate Count

Measurements > Estimate Count

Estimates number of features as the total area of selected pixels in a layer divided by the average area of features in a chosen Memory Image. Essentially, it determines how many average features from the Relative To selection would fit within the layer selection. Works best for features which are difficult to separate, and which are all similar in size. Several representative features of interest should be selected in the chosen Memory Image to provide the most accurate estimated count



1. Measure Relative To

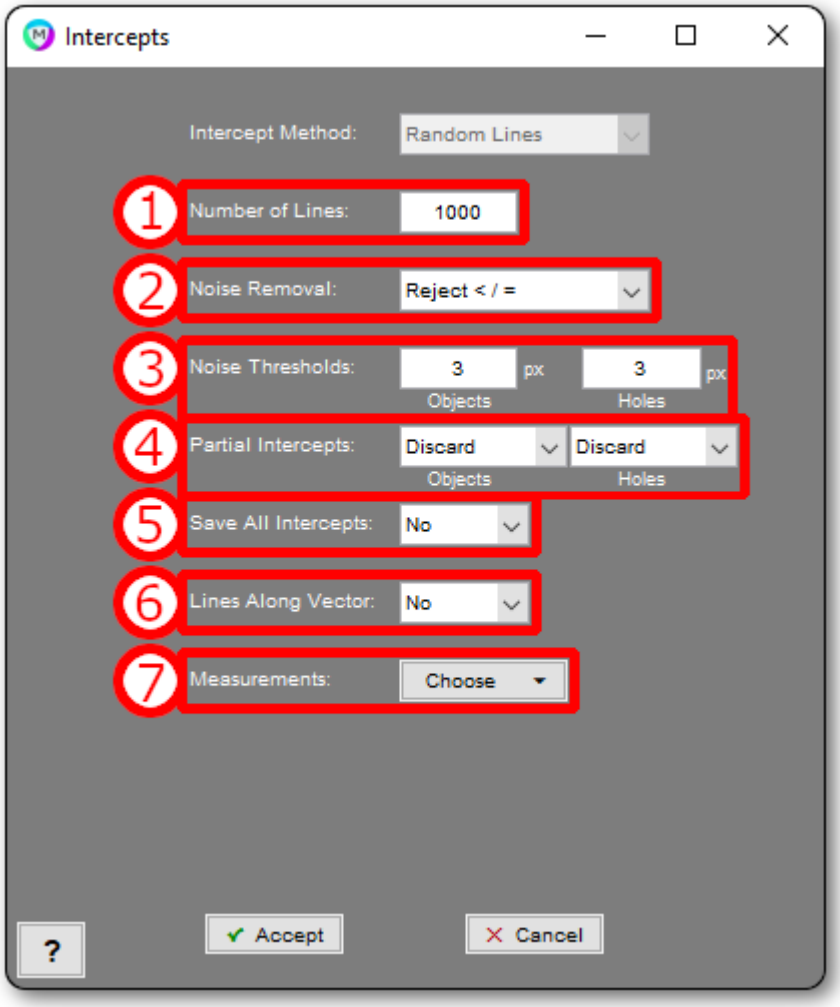
Choose which Memory Image to use to estimate the number of features. The total selected area in the image of interest is divided by the average feature area in the chosen Memory Image.

Intercepts

Measurements > Intercepts

Random Lines

Measures metrics such as mean intercept by drawing a specified number of random or rotating lines through the features in the Current Image.



1. Number of Lines

Number of lines to draw

2. Noise Removal

Method for removing small intercepts

- **Reject < / =:** Simply ignores intercepts below a certain length
- **Fit to Noise:** *(Only for "Random Lines")* Fits an exponential to the first few bins and subtracts this exponential out of the intercept histogram.

3. Noise Thresholds

Thresholds for either the minimum intercept length (for "Reject </="), the number of bins to use for "Fit to Noise"

4. Partial Intercepts

Choose whether to discard or keep intercepts that start or end at an image edge

5. Save All Intercepts

Choose whether to save all intercept lengths into text files

6. Lines Along Vector

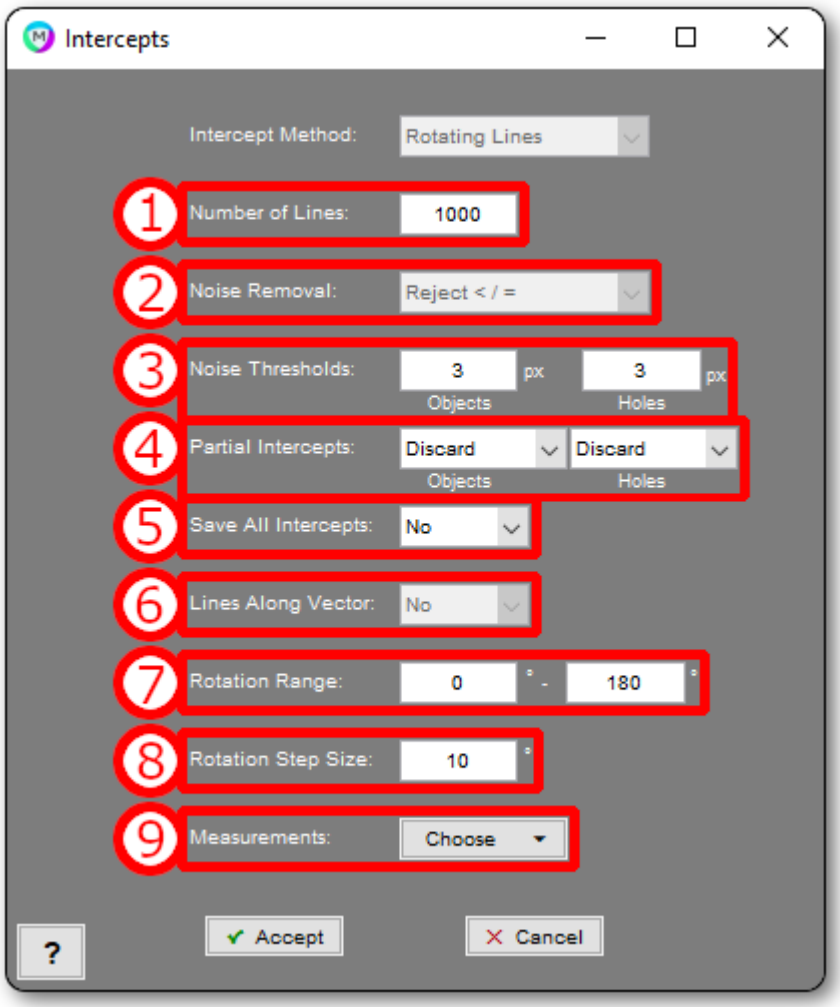
Choose whether to draw all "Random Lines" along a specific direction. Starting points of each line will still be random.

7. Measurements

Choose which measurements to report (described below).

Rotating Lines

Measures metrics such as mean intercept by rotating a grid of specified number of lines through the features in the Current Image.



1. Number of Lines

Number of lines to draw

2. Noise Removal

Method for removing small intercepts

- **Reject < / =:** Simply ignores intercepts below a certain length
- **Fit to Noise:** *(Only for "Random Lines")* Fits an exponential to the first few bins and subtracts this exponential out the intercept histogram.

3. Noise Thresholds

Thresholds for either the minimum intercept length (for "Reject </="), the number of bins to use for "Fit to Noise"

4. Partial Intercepts

Choose whether to discard or keep intercepts that start or end at an image edge

5. Save All Intercepts

Choose whether to save all intercept lengths into text files

6. Lines Along Vector

Disabled in “Rotating Lines” mode

7. Rotation Range

Range of angles through which to rotate the grid of parallel lines

8. Rotation Step Size

Increment with which to step though angle range to rotate the grid of parallel lines

9. Measurements

Choose which measurements to report (described below).

Measurements

- **Mean Intercept – Objects:** Average length of intercepts which pass through selected features (objects)
- **Mean Intercept – Holes:** Average length of intercepts which pass through un-selected areas (holes)
- **Mean Inverse Intercept – Objects:** Average inverse length of intercepts which pass through selected features (objects)
- **Mean Inverse Intercept – Holes:** Average inverse length of intercepts which pass through un-selected areas (holes)
- **Mode Intercept – Objects:** Mode (most common) length of intercepts which pass through selected features (objects)
- **Mode Intercept – Holes:** Mode (most common) length of intercepts which pass through un-selected areas (holes)
- **Mode Inverse Intercept – Objects:** Mode (most common) inverse length of intercepts which pass through selected features (objects)
- **Mode Inverse Intercept – Holes:** Mode (most common) inverse length of intercepts which pass through un-selected areas (holes)
- **ASTM Grain Size Number:** ASTM standard grain size number derived from mean intercept length through selected features (objects)

ASTM number is only produced from random line measurements, is only valid for grain size measurements, and is only produced when a calibration factor is present. It is calculated using the equation: *ASTM = 10-2*log2([Mean Int Black in Microns]/10)* [1]

- **Total Line Length:** Total length of all lines which are drawn in order to collect intercepts.
- **Total Intersections:** Number of intersections produced between the drawn lines and intercepted features.

References

[1] ASTM E112-13, Standard Test Methods for Determining Average Grain Size, ASTM International, West Conshohocken, PA, 2013, www.astm.org

Image Dimensions

Measurements > Image Dimensions

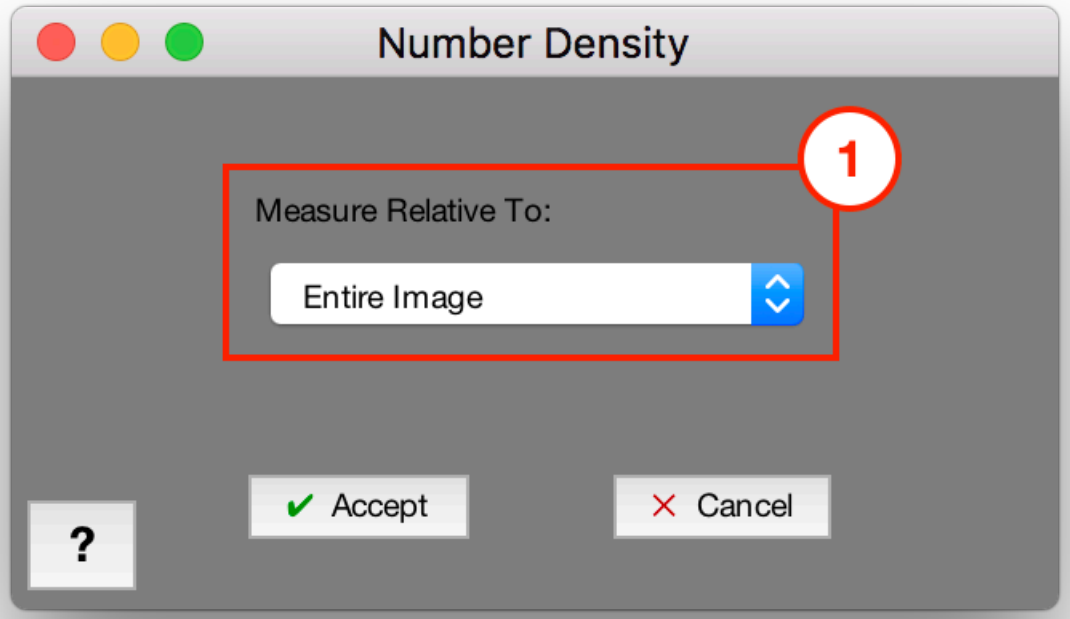
Measures the X and Y dimensions of the Current Image.

May be performed on grayscale Current Images.

Number Density

Measurements > Number Density

Measures the total number of separate features per area in the Current Image, or per selection area in a chosen Memory Image.



1. Measure Relative To

Choose to measure relative to either the entire image, or the selection in a Memory Image

Perimeter

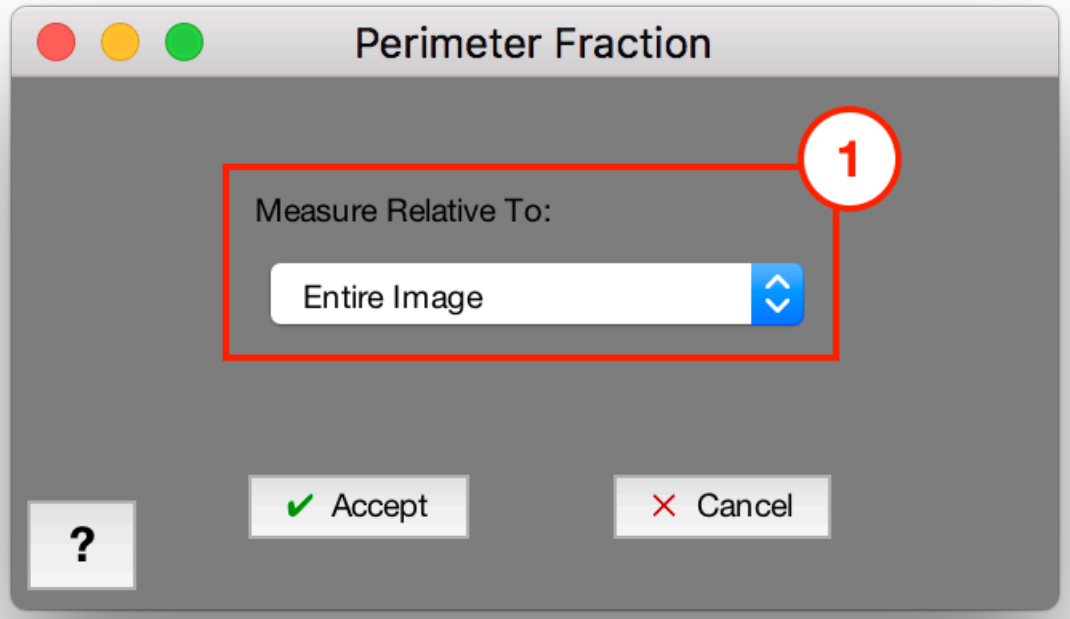
Measurements > Perimeter

Measures the total perimeter length around all features in the current image.

Perimeter Fraction

Measurements > Perimeter Fraction

Measures the total perimeter fraction of features in the current image. This is calculated as the total perimeter length divided by the either the total number of pixels in the image, or the total number of selected pixels in a chosen Memory Image.



1. Measure Relative To

Choose to measure relative to either the entire image, or the selection in a Memory Image

*Intensity Mean

Measurements > *Intensity Mean

Requires Companion Image

Measures the average grayscale intensity within selection based on latest Companion Image

MIPAR will generate the intensity measurement using the original image bit depth, see Companion Image for a list of recipe steps that maintain bit depth.

*Intensity StdDev

Measurements > *Intensity StdDev

Requires Companion Image

Measures the standard deviation of grayscale intensity within selection based on latest Companion Image

MIPAR will generate the intensity measurement using the original image bit depth, see Companion Image for a list of recipe steps that maintain bit depth.

*Intensity Sum

Measurements > *Intensity Sum

Requires Companion Image

Measures the total of grayscale intensity within selection based on latest Companion Image

MIPAR will generate the intensity measurement using the original image bit depth, see Companion Image for a list of recipe steps that maintain bit depth.

*Correlation Coefficient

Measurements > *Correlation Coefficient

Requires Companion Image

Measures the normalized and absolute correlation coefficient [1-3] between the Current and Companion images. The normalized value is a measure of image similarity that well-suited for images of similar intensity types.

May be performed on grayscale Current Images.

References

[1] Fisher, R.A. Statistical Methods for Research Workers, 13th Ed., Hafner, 1958.

[2] Kendall, M.G. The Advanced Theory of Statistics, 4th Ed., Macmillan, 1979.

[3] Press, W.H., Teukolsky, S.A., Vetterling, W.T., and Flannery, B.P. Numerical Recipes in C, 2nd Ed., Cambridge University Press, 1992.

*Mutual Information

Measurements > *Mutual Information

Requires Companion Image

Measures the normalized and classical (absolute) amount of mutual information [1,2] between the Current and Companion images. The normalized value is a measure of image similarity that is well-suited for images of both similar and different intensity types.

May be performed on grayscale Current Images.

References

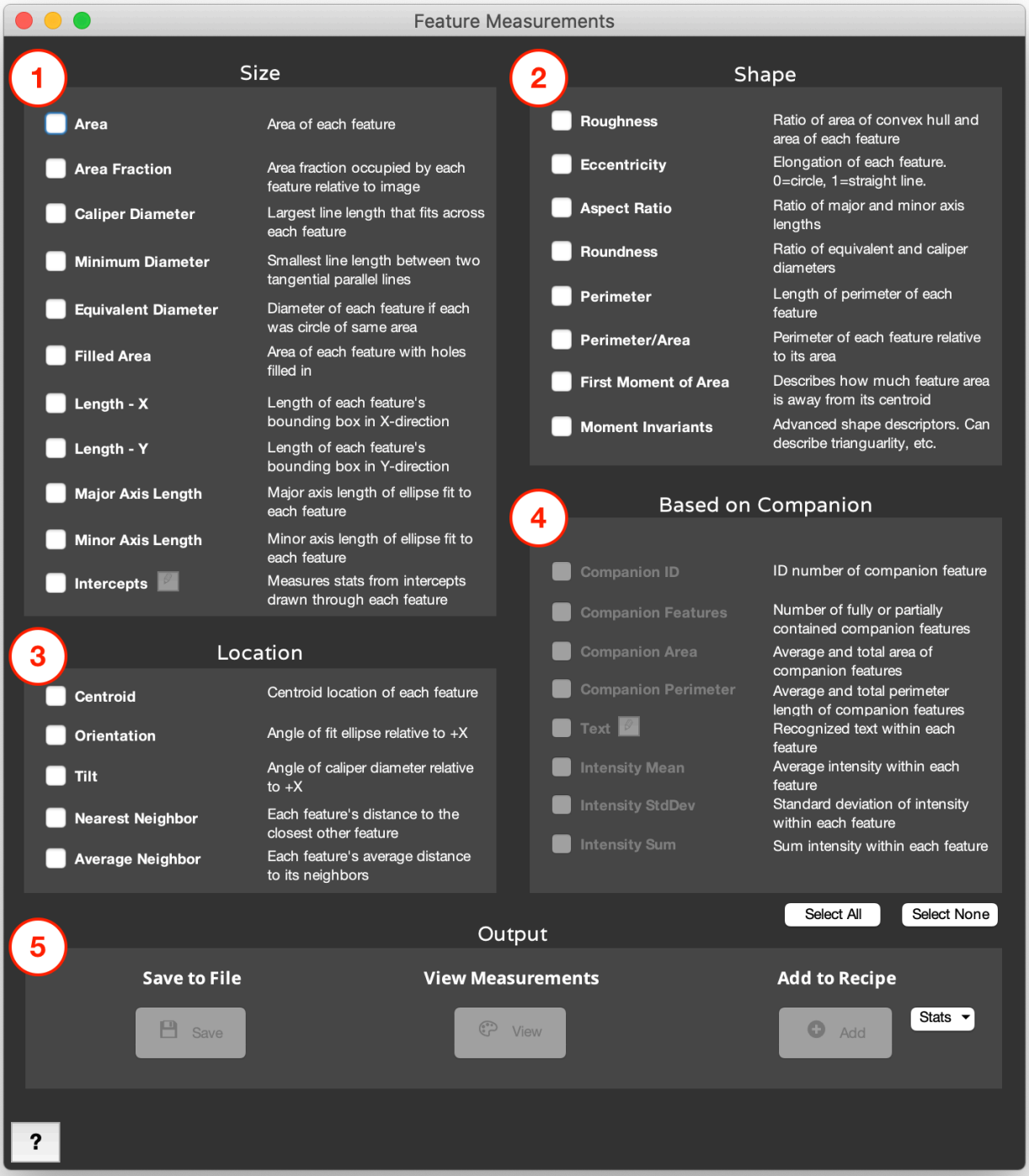
[1] Rahunathan, Smriti, D. Stredney, P. Schmalbrock, and B.D. Clymer. Image Registration Using Rigid Registration and Maximization of Mutual Information. Poster presented at: MMVR13. The 13th Annual Medicine Meets Virtual Reality Conference; 2005 January 26–29; Long Beach, CA.

[2] D. Mattes, D.R. Haynor, H. Vesselle, T. Lewellen, and W. Eubank. “Non-rigid multimodality image registration.” (Proceedings paper).Medical Imaging 2001: Image Processing. SPIE Publications, 3 July 2001. pp. 1609–1620.

Feature Measurements

Measurements > Feature Measurements

Feature Measurements allows a variety of measurements to be made from each feature in the image or Layer.



1. Size

- **Area**: Area of each feature.
- **Area Fraction**: Area fraction occupied by each feature relative to entire image.
- **Caliper Diameter**: Largest line length that fits within each feature.
- **Minimum Diameter**: Smallest line length that fits between two parallel lines tangential to each

feature.

- **Equivalent Diameter**: Diameter of each feature if each was a circle of the same area.
- **Filled Area**: Area of each feature with holes filled in.
- **Length – X**: Length of each feature's bounding box in the X-direction.
- **Length – Y**: Length of each feature's bounding box in the Y-direction.
- **Major Axis Length**: Major axis length of ellipse fit to each feature [1].
- **Minor Axis Length**: Minor axis length of ellipse fit to each feature [1].
- **Intercepts**: Measures statistics from intercept length recorded from each feature.
  - **Through Centroid**: Draw each line through each feature's centroid.
  - **Along Vector**: Draw each line along the same direction through each feature.

2. Shape

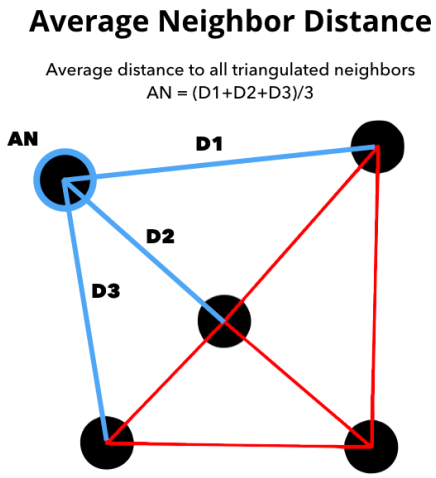
- **Roughness**: Ratio of area of tightest-fitting convex hull and area of each feature.
- **Eccentricity**: Describes how elongated or circular each feature is. 0 is a perfect circle. 1 is a straight line. Eccentricity is calculated from the ellipse fit to each feature as *sqrt(-1([minor axis length]/[major axis length])^2)* [1])
- **Aspect Ratio**: Ratio of major and minor axis lengths, as defined by the ellipse fit to each feature [1].
- **Roundness**: Ratio of equivalent and caliper diameters.
- **Perimeter**: Length of perimeter of each feature.
- **Perimeter/Area**: Perimeter of each feature relative to its area.
- **First Moment of Area**: Describes how much area is extended away from each feature's centroid. Calculated as the summation of (area x distance from centroid) over all pixels in the feature.
- **Moment Invariants**: High-order moments which describe different shape properties of each feature [2,3].

ROUNDNESS	HIGH	LOW	LOW
ECCENTRICITY	LOW	LOW	HIGH
ROUGHNESS	LOW	HIGH	LOW

3. Location

- **Centroid**: Centroid location of each feature.
- **Orientation**: Angle of the ellipse fit to each feature with respect to the positive X-axis. Positive angles

are clockwise rotations and negative counterclockwise.
- **Tilt**: Angle of the caliper diameter of each feature with respect to the positive X-axis. Positive angles are clockwise rotations and negative counterclockwise.
- **Nearest Neighbor**: Each feature's distance to the closest other feature, calculated from the features' centroids.
- **Average Neighbor**: Each feature's average distance to its neighbors, as defined by the features' Delaunay triangulation. Triangulation and distance are both calculated from the features' centroids.



4. Based on Companion

Requires Companion Image Each layer's Companion Image is determined independently from the recipe, as the latest Companion Image set before the layer step. MIPAR will generate the intensity measurement using the original image bit depth, see Companion Image for a list of recipe steps that maintain bit depth.

- **Companion ID**: ID number of the feature in the Companion Image that contains the centroid of each feature being measured.
- **Companion Features**: Number of features in the Companion Image which are within or in contact with each feature in the Current Image.
- **Companion Area**: Average, total, and fractional area of features in the Companion Image within each feature in the Current Image.
- **Companion Perimeter**: Average and total perimeter length of features in the Companion Image within each feature in the Current Image.
- **Text**: Recognizes text within each feature. Set the binary mask or grayscale image as the companion. The Current Image should the the bounding box of the text, with the major axis running parallel to the text orientation.
  - **Edit**: Allows the text search to be limited to just numbers
  - Sample Image Sample Recipe
- **Intensity Mean**: Average grayscale intensity in the latest Companion Image within each feature in the Current Image. Select the bit-depth from the dropdown.
- **Intensity StdDev**: Standard deviation of grayscale intensity in the latest Companion Image within each feature in the Current Image. Select the bit-depth from the dropdown.

- **Intensity Sum**: Sum grayscale intensity in the latest Companion Image within each feature in the Current Image. Select the bit-depth from the dropdown.

5. Output

- **Save to File**: Saves the selected feature measurements to a tab-delimited .txt file.
- **View Measurements**: View the selected measurements as a list, histogram, and image overlay (in *Color by Measurements* only). For measurements made on one layer, they will be viewed in our *Color by Measurements* tool, while measurements made on several or all layers will be viewed in our *Histogram of Measurements* tool.
- **Add to Recipe**: Adds the selected summary statistics to the recipe. Options are Mean, Min, Max, Median, Standard Deviation, and Sum.

References

[1] Ellipse is fit to the feature as the ellipse with the same second moments as the feature. Second moments are calculated as:
uxx = sum(x^2)/N + 1/12
uyy = sum(y^2)/N + 1/12
*uxy = sum(x*y)/N*

where x is the x coordinate of each pixel in the feature, y is the y coordinate of each pixel, and N is the number of pixels.

The major and minor axes are calculated as:

*common = sqrt((uxx – uyy)^2 + 4*uxy^2)*
*MajorAxisLength = 2*sqrt(2)*sqrt(uxx + uyy + common)*
*MinorAxisLength = 2*sqrt(2)*sqrt(uxx + uyy – common)*

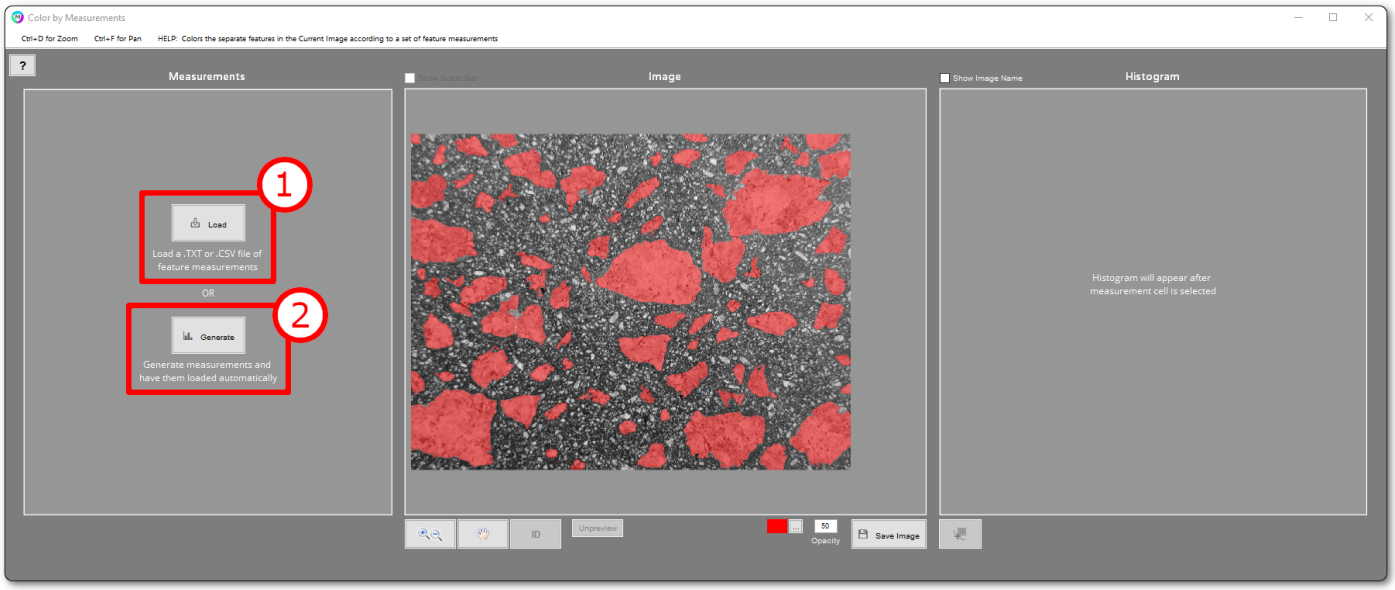
[2] Rosin, Paul L. 2003. “Measuring Shape: Ellipticity, Rectangularity, and Triangularity.” Machine Vision and Applications 14 (3): 172–184.

[3] MacSleyne, J. P., Simmons, J. P., & De Graef, M. (2008). On the use of 2-D moment invariants for the automated classification of particle shapes. Acta Materialia, 56(3), 427–437.

Color by Measurements

Measurements > Color By Measurements

Colors the separate features in the Current Image according to a set of feature measurements.

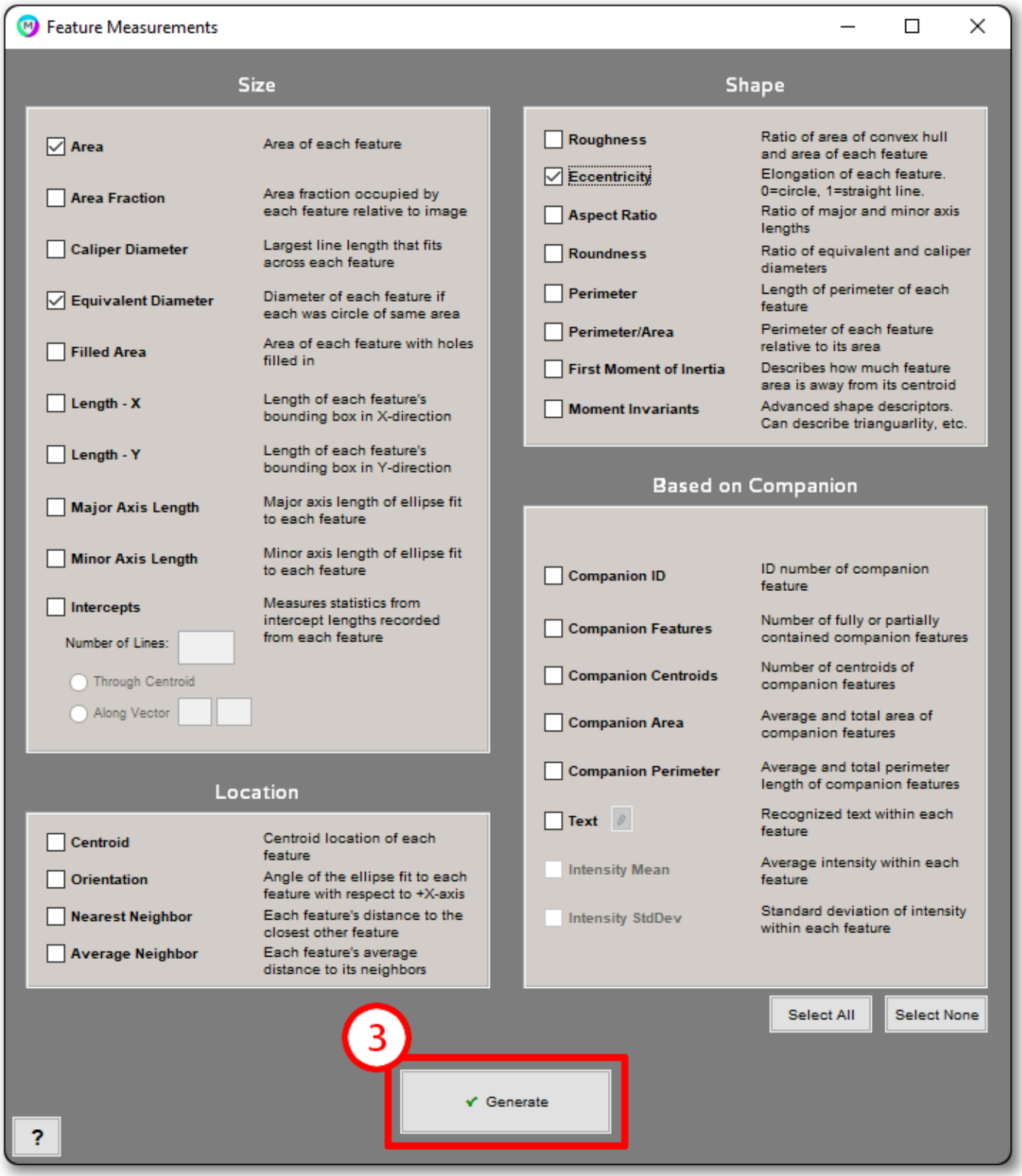


1. Load

- If you have already generated measurements beforehand and then decide to produce a color-coded image, you can load the measurements file here.

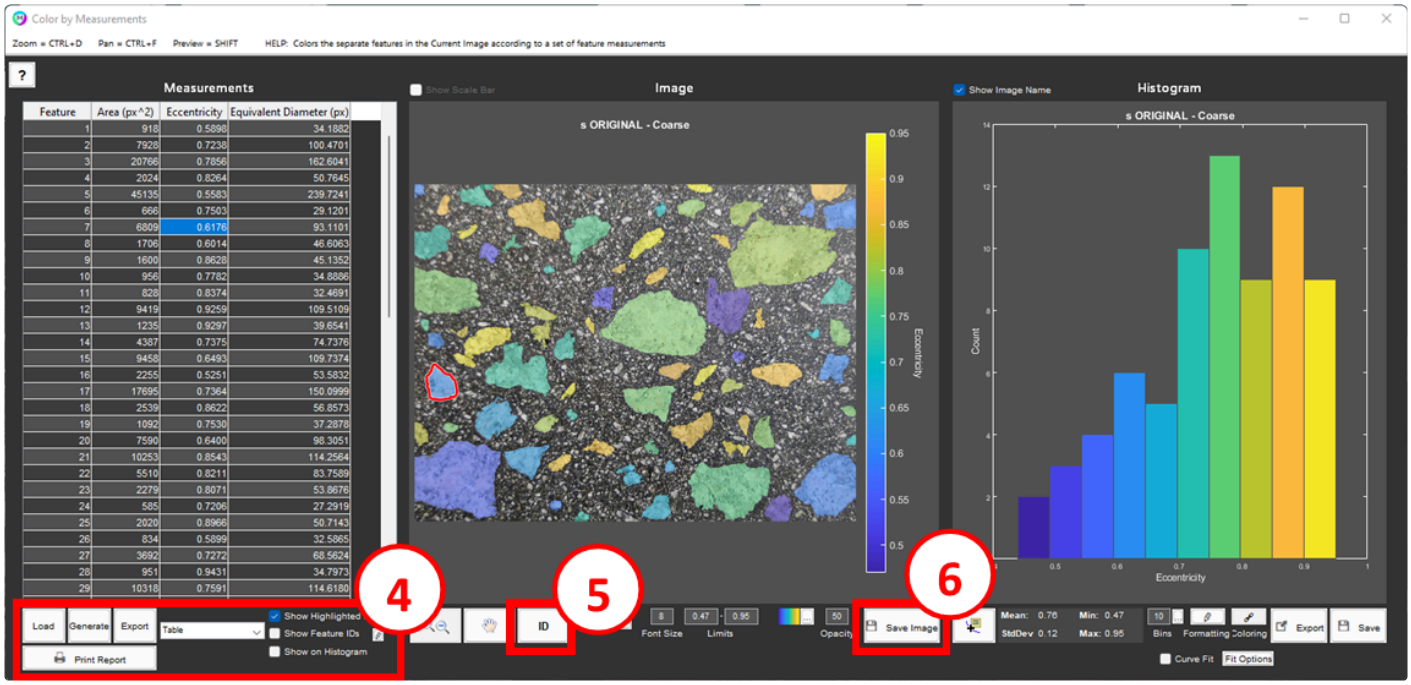
2. Generate

- If you have not yet generated measurements, you can generate measurements here to produce a color-coded image. Then, the following window will appear where you can select your desired feature measurements for your color-coded image. Refer to Feature Measurements page for further description on each feature measurement.



3. Generate

Perform the selected measurements and return to the Color by Measure window.



4. Load, Generate, Export

- **Load**: You can start over and create a new color-coded image by loading a new feature measurements file.
- **Generate**: You can create a new color-coded image by selecting new feature measurements.
- **Export**: You can export the existing feature measurements results to a spreadsheet, or to local measurements using a sliding window.
- **Show Highlighted Feature**: Outlines the feature corresponding to the highlighted cell.
- **Show Feature IDs**: Places the Feature Number next to each feature. Select the Edit button to limit the percentile displayed, change the text font size and color.
- **Show on Histogram**: Places a marker on the histogram for the highlighted cell.

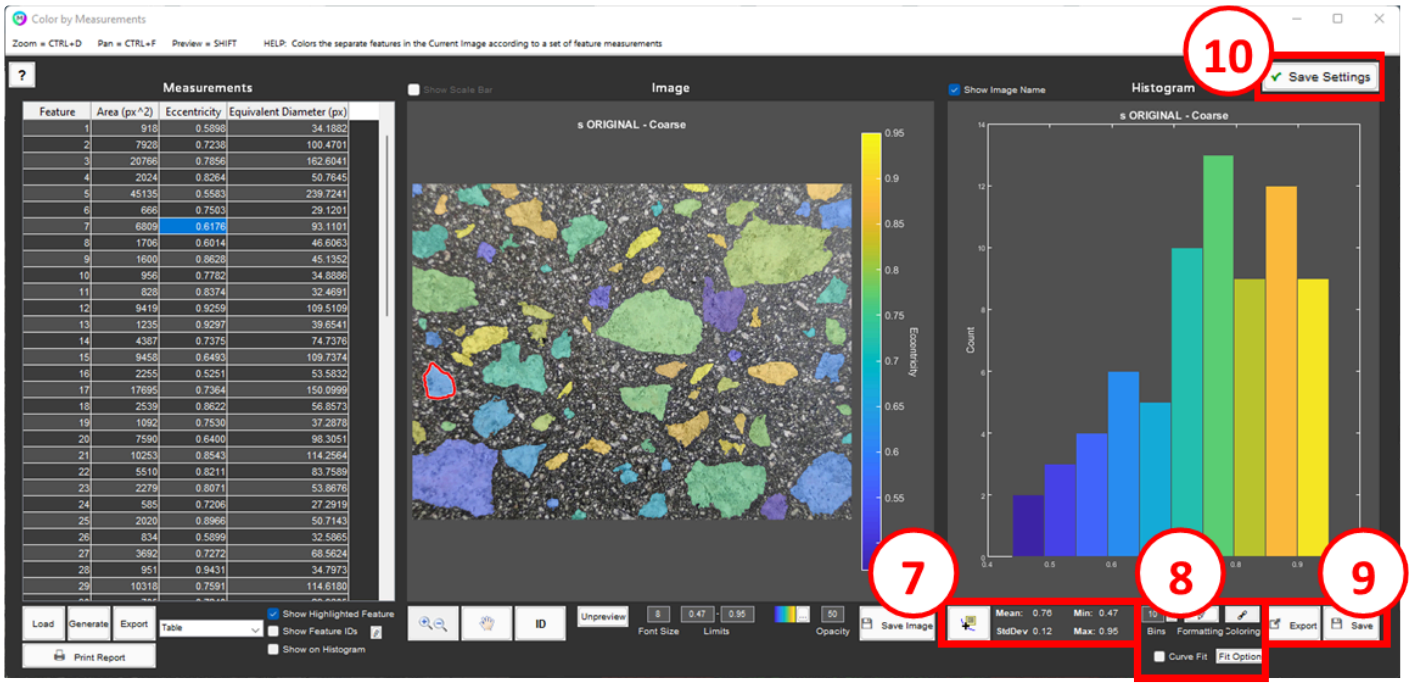
Click on any cell in the column you desire to create the corresponding color-coded image.

5. ID

Click a feature to identify its corresponding measurement in table

6. Save Image

Save Image allows you to save the current “color by measurements” image.



7. Measurement Stats

Displays the mean, standard deviation, min, and max of the selected measurements.

> Right-click on these stats and select “Copy” to copy them to the clipboard

8. Histogram Display Settings

- Bins determines the number of bins to divide the histogram into. The default value is auto-calculated to provide an effective histogram shape. Typing in a new value sets a manual number of bins. Erase the typed in value anytime to revert back to the auto-calculated value. Click the ellipsis box next to “Bins” to manually set the limits of each bin.
- Click the pencil icon to show the X- and Y-axis limits, which can be edited by entering in the desired values.
- Coloring controls the color of histogram bars, either to be linked to the chosen colormap or to all be of a single chosen color.
- Add a Curve Fit by selecting any of the 3 fit types: Normal, Lognormal and Bimodal. Fit parameters are displayed by selecting the Fit Parameters button. These are automatically added to the Report Generator.

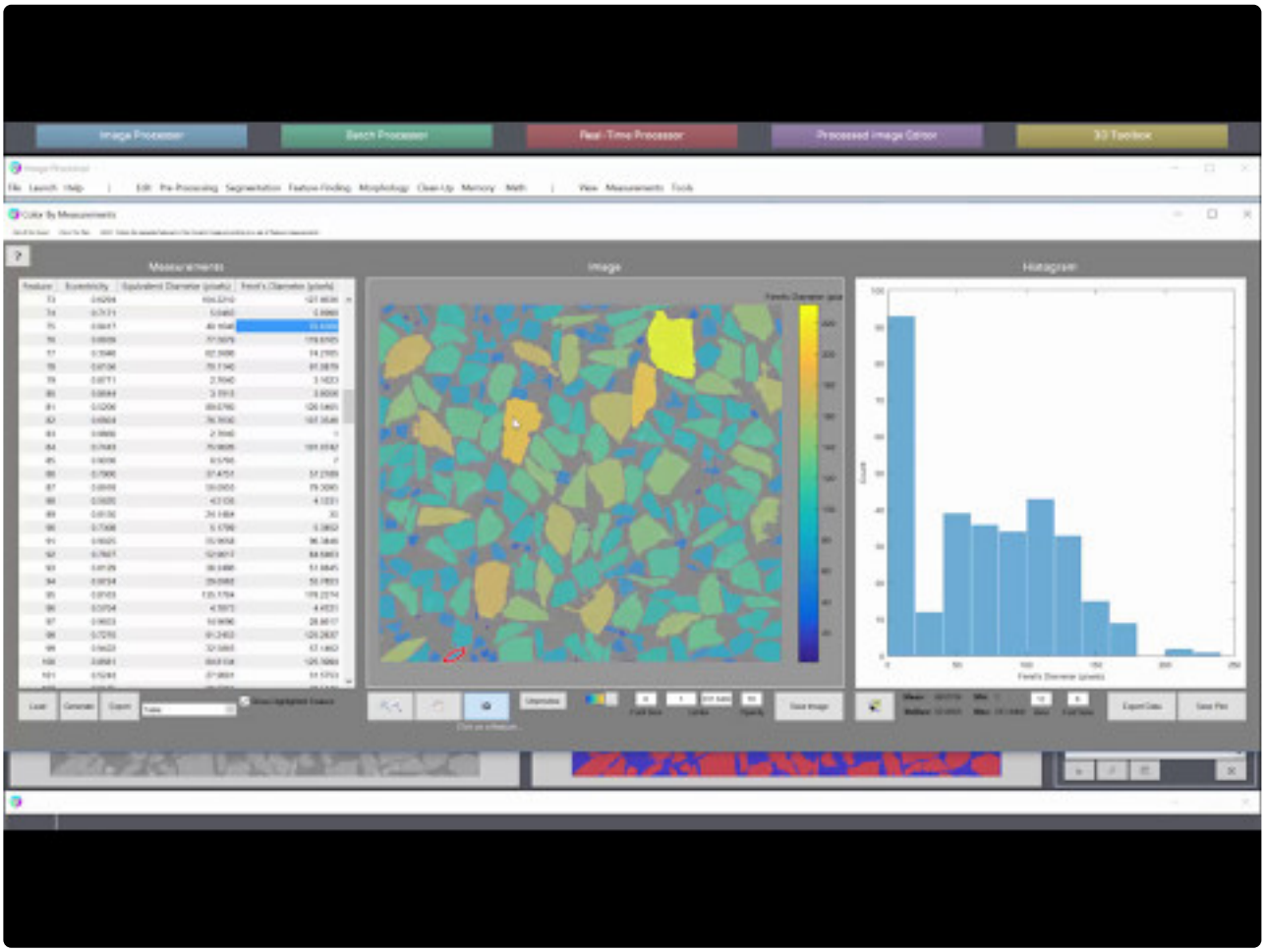
9. Export Data, Save Plot

- **Export Data**: Export histogram data to spreadsheet.
- **Save Plot**: Save plot as image.

10. Save Settings

If Color by Measurements is reached through the play button, then adjustments to display settings, such as selected column, image limits and coloring and histogram bins, limits, and coloring, will be saved into the recipe. The next time Color by Measurements is run on this measurement, these settings will be automatically used.

Tutorial



https://www.youtube.com/embed/kvhZrzKOubE?rel=0

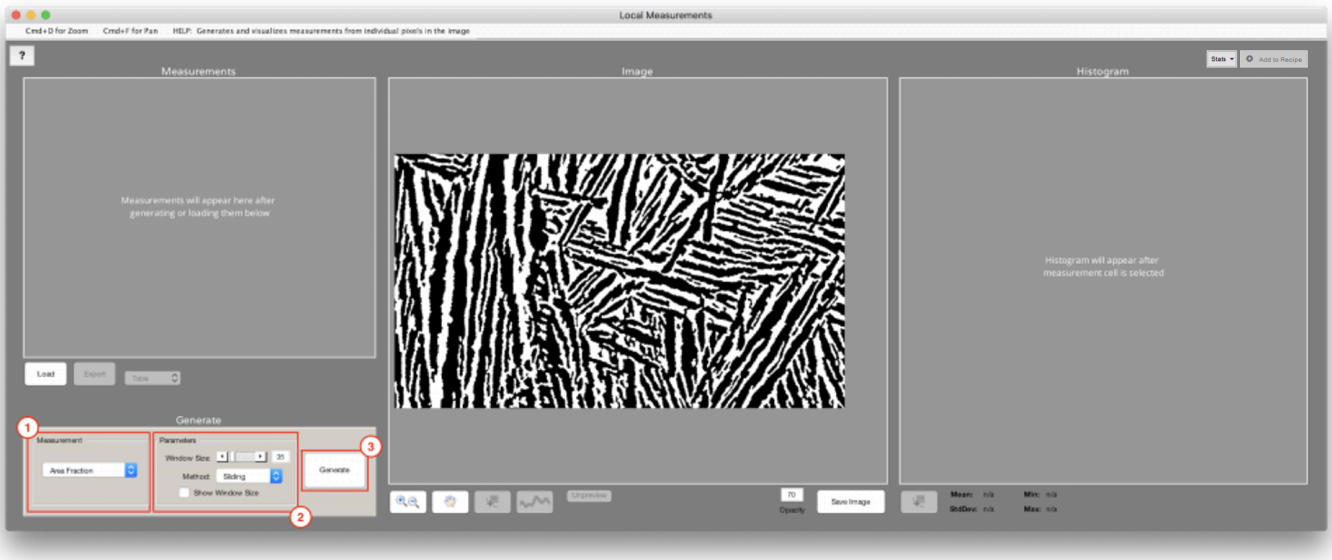
https://www.youtube.com/embed/kvhZrzKOubE?rel=0

https://www.youtube.com/embed/kvhZrzKOubE?rel=0

Local Measurements

Measurements > Local Measurements

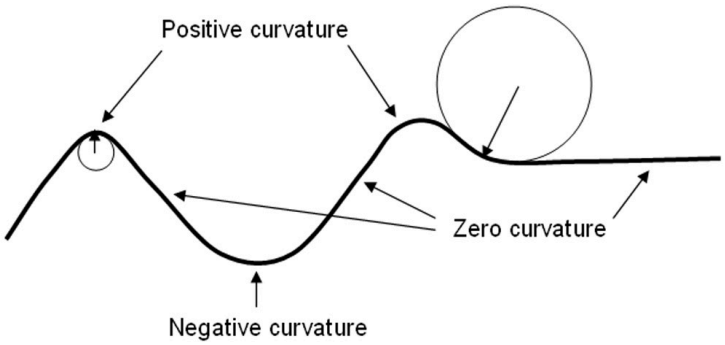
Generates and visualizes measurements from individual pixels in the image.



1. Measurement

Select the following types of measurements you would like to see visualized in the image.

- **Anisotropy**: Measures the amount of local directionality of features at each pixel in the image, with 0 being completely random in direction and 1 being highly directional
- **Area Fraction**: Measures the local area fraction of selected features at each pixel in the image
  - **Step Size**: Determines the steps to take between measured pixels. Measurements for pixels in between are interpolated
- **Count**: Measures the number of features around each pixel
- **Curvature**: Generates the local curvature of the feature perimeter by fitting a circle at each pixel. [1]
  - **Smoothing**: The smoothing determines the “search radius” (in pixels) about each pixel on the perimeter which is used to find pixels to fit a circle to. Larger smoothing values will results in more gradual transitions between curvatures along the perimeter, but may prevent very high curvatures from being measured properly.
  - **Method**: Local curvature measurements are made at each pixel along the features’ skeleton. The algorithm which calculates this skeleton can be selected here.
    - **Perimeter**: Local curvature measurement is made along the perimeter of the features
    - **Skeleton**: Local curvature measurement is made along the skeleton of the features



- **Number Density**: Measures the number of features per unit area around each pixel
- **Thickness**: Measures the local thickness along features by measuring the diameter of the largest circle that can fit inside a feature with the center of the circle at each point along the feature's skeleton
    - **Trim Thresh**: The trim threshold determines the minimum length (in pixels) that all skeleton branches must be. Any branches below this length will be removed. This can be useful for removing small branches in the corners of features which do not represent pixels that one wishes to measure for thickness. Setting this threshold will cause the skeleton to be adjusted; however, the measurements themselves will not be re-calculated until the user clicks "Generate".
    - **Method**: Local thickness measurements are made at each pixel along the features' skeleton. The algorithm which calculates this skeleton can be selected here.
        - **Advanced**: Uses morphological thinning to produce skeletons typically with fewer branches. Often leads to more accurate thickness measurements.
        - **Smooth**: Uses morphological thinning to produce skeletons typically with very few branches. Generates continuous skeletons best used for curved features.
        - **Classic**: Uses traditional skeletonization to produce skeletons typically with more branches. However, in the case of very simple shapes, this method can produce slightly longer skeletons such that more desired pixels are measured than with the advanced method.
        - **Fit to Major**: Draws lines along the major axes of all features. This is the recommend method for measuring local thickness, parallel to the thinnest dimension, along a single, or a just few, feature(s) (e.g., a layer of some sort) in the image. This method avoids branches that can occur near image edges when using the "Advanced" and "Classic" methods.
        - **Fit to Minor**: Draws lines along the minor axes of all features. This is the recommend method for measuring local thickness, parallel to the thickest dimension, along a single, or a just few, feature(s) (e.g., a layer of some sort) in the image. This method avoids branches that can occur near image edges when using the "Advanced" and "Classic" methods.
- **Orientation**: Measures the local orientation of features at each pixel in the image from – 90 degrees to + 90 degrees
- * **Intensity Mean**: *Requires that a grayscale Companion Image be set earlier in the Recipe.* Measures the local average grayscale intensity at each pixel within the features. The measured grayscale

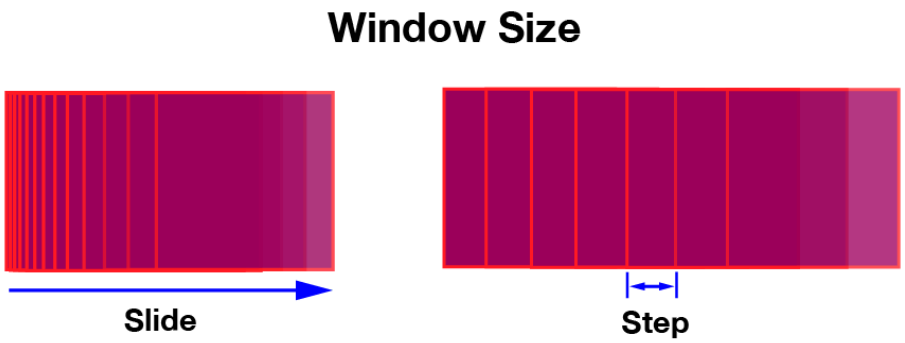
intensities come from the latest set Companion Image.

- * **Intensity StdDev**: *Requires that a grayscale Companion Image be set earlier in the Recipe.* Measures the local standard deviation in grayscale intensity at each pixel within the features. The measured grayscale intensities come from the latest set Companion Image.
- * **Nearest Distance**: *Requires that a B/W Companion Image be set earlier in the Recipe.* Measures the the distance between each feature's pixel and the nearest feature pixel from the latest set Companion Image.

Shared Parameters

- **Window Size**: For ***Anisotropy***, ***Area Fraction***, ***Count***, ***Number Density***, ***Orientation***, ***Intensity Mean***, and ***Intensity StdDev*** the window size can be adjusted. The window size determines the number of pixels that will be used to calculate a measurement value for each pixel.

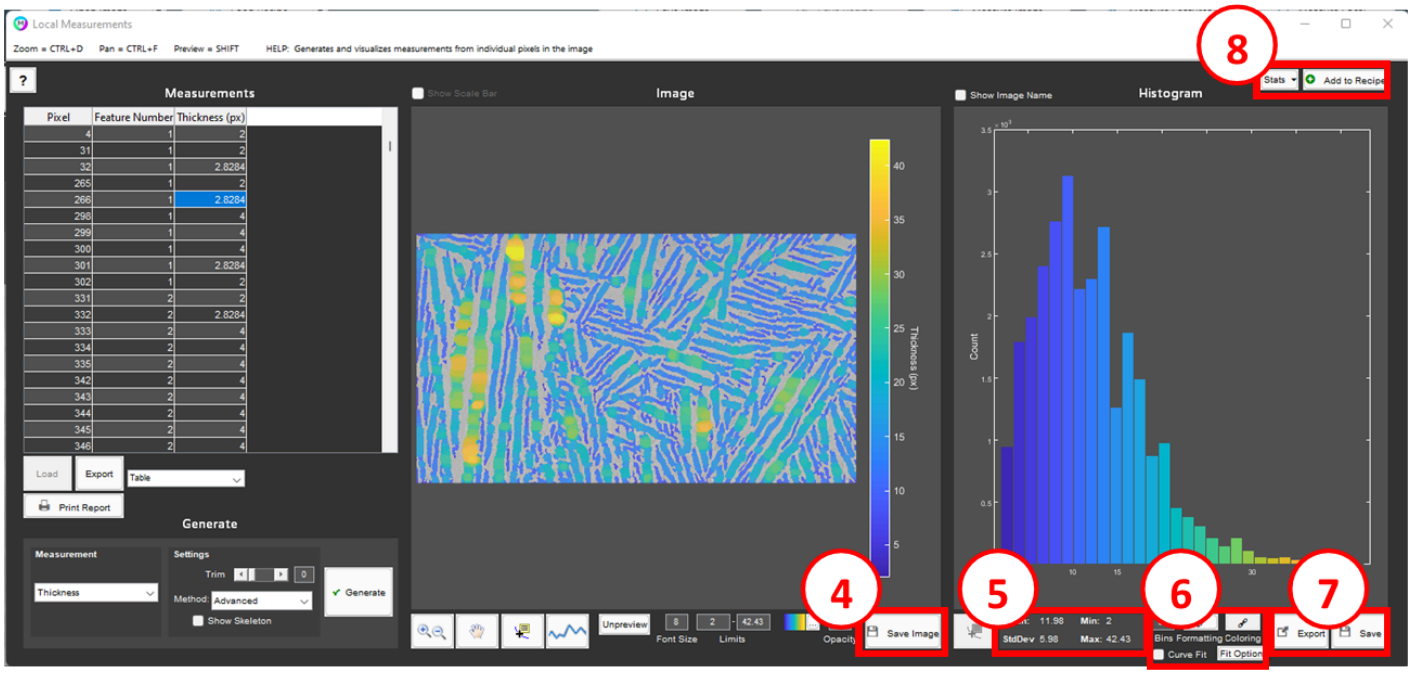
- **Method**: For ,***Area Fraction***, ***Count***, ***Number Density***, ***Intensity Mean***, and ***Intensity StdDev*** the method can be selected. Sets the method for window translation across the image. Options are ***Sliding (Slide)*** or ***Blocks (Step)***.



- **Speed**: For ***Orientation*** and ***Anisotropy*** the speed can be adjusted. Speed controls the speed vs precision tradeoff. Higher speed settings use more downsampling on the image prior to map calculation.

3. Generate

- Generate measurements with the current settings and print them in the table on the left



4. Save Image

- Save image anywhere on your computer.

5. Measurement Stats

Displays the mean, standard deviation, min, and max of the selected measurements.

> Right-click on these stats and select “Copy” to copy them to the clipboard

6. Histogram Parameters

- Determines the number of bins to divide the histogram into. The default value is auto-calculated to provide an effective histogram shape. Typing in a new value sets a manual number of bins. Erase the typed in value anytime to revert back to the auto-calculated value.
- Edit bin limits, font size etc
- Add a Curve Fit by selecting any of the 3 fit types: Normal, Lognormal and Bimodal. Fit parameters are displayed by selecting the Fit Parameters button. These are automatically added to the Report Generator.

7. Export Data, Save Plot

- **Export Data**: Export histogram data to spreadsheet.
- **Save Plot**: Save plot as image.

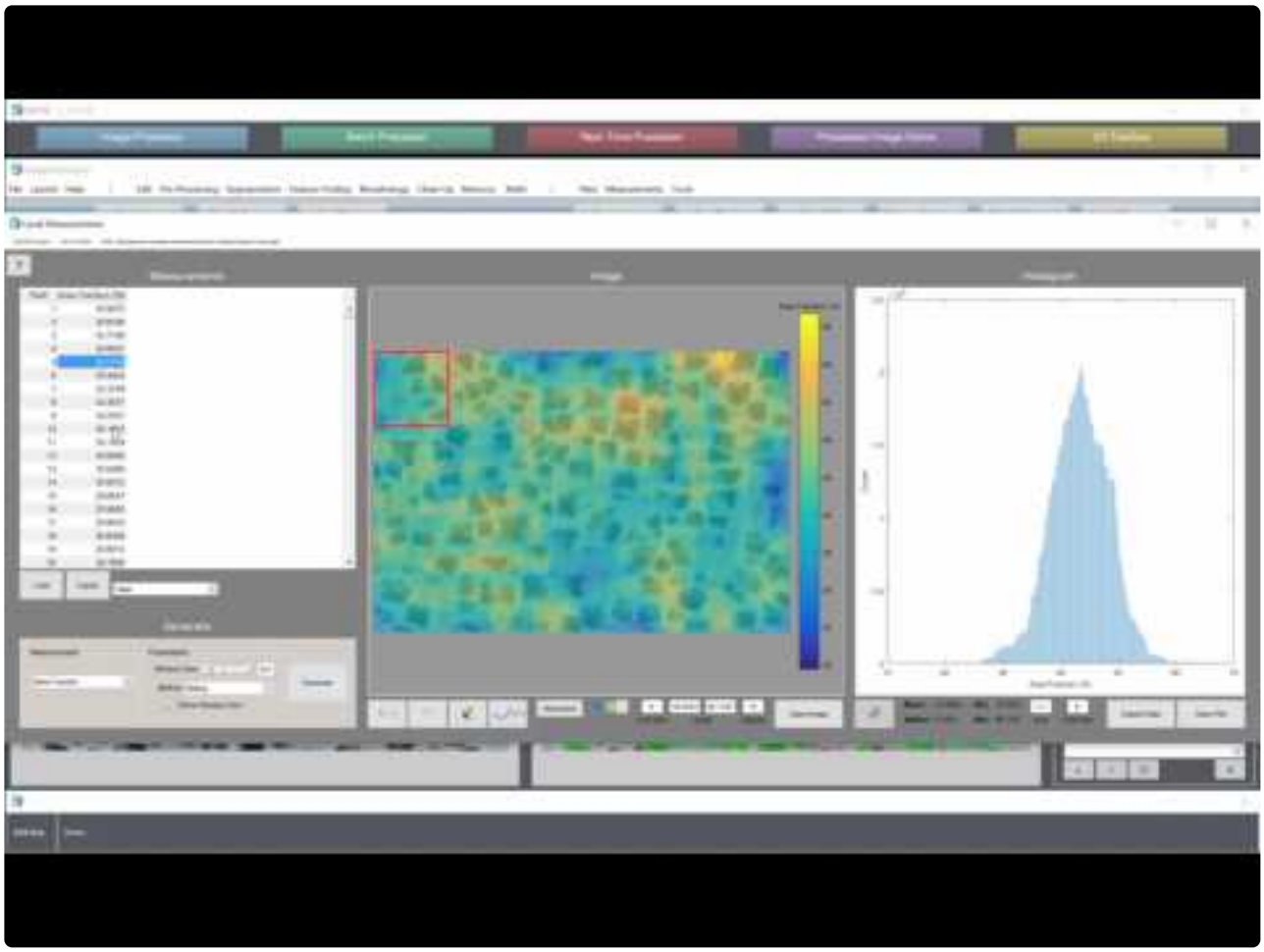
8. Add to Recipe

- **Add to Recipe**: Add local measurement as step to recipe. Measurement and format settings will be preserved and recalled when step is played from recipe.
- **Stats**: Summary statistics to add to the recipe, which will appear in the global measurement report. Options are Mean, Min, Max, Median, Standard Deviation, and Sum.

References

1. Bravo, L., & Aldana, M. (2010). Volume curvature attributes to identify subtle faults and fractures in carbonate reservoirs: Cimarrona Formation, Middle Magdalena Valley Basin, Colombia. SEG Technical Program Expanded Abstracts 2010. doi:10.1190/1.3513293

Tutorial



https://www.youtube.com/embed/WhAKluBliXs?rel=0

https://www.youtube.com/embed/WhAKluBliXs?rel=0

https://www.youtube.com/embed/WhAKluBliXs?rel=0

Tools

Contains miscellaneous tools for handling data output by MIPAR functions.

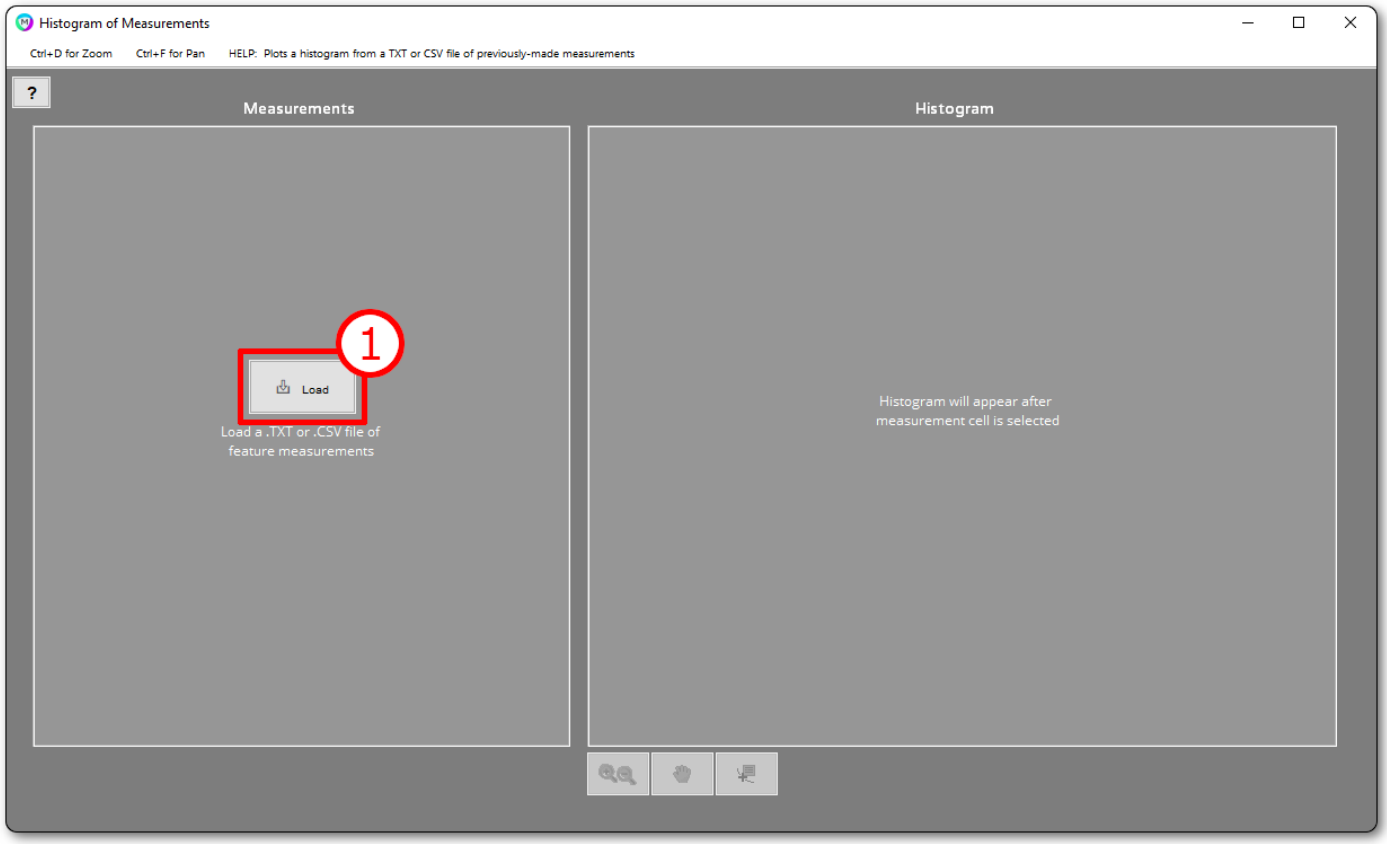
Functions

- Histogram of Measurements
- Construct Polar Plot

Histogram of Measurements

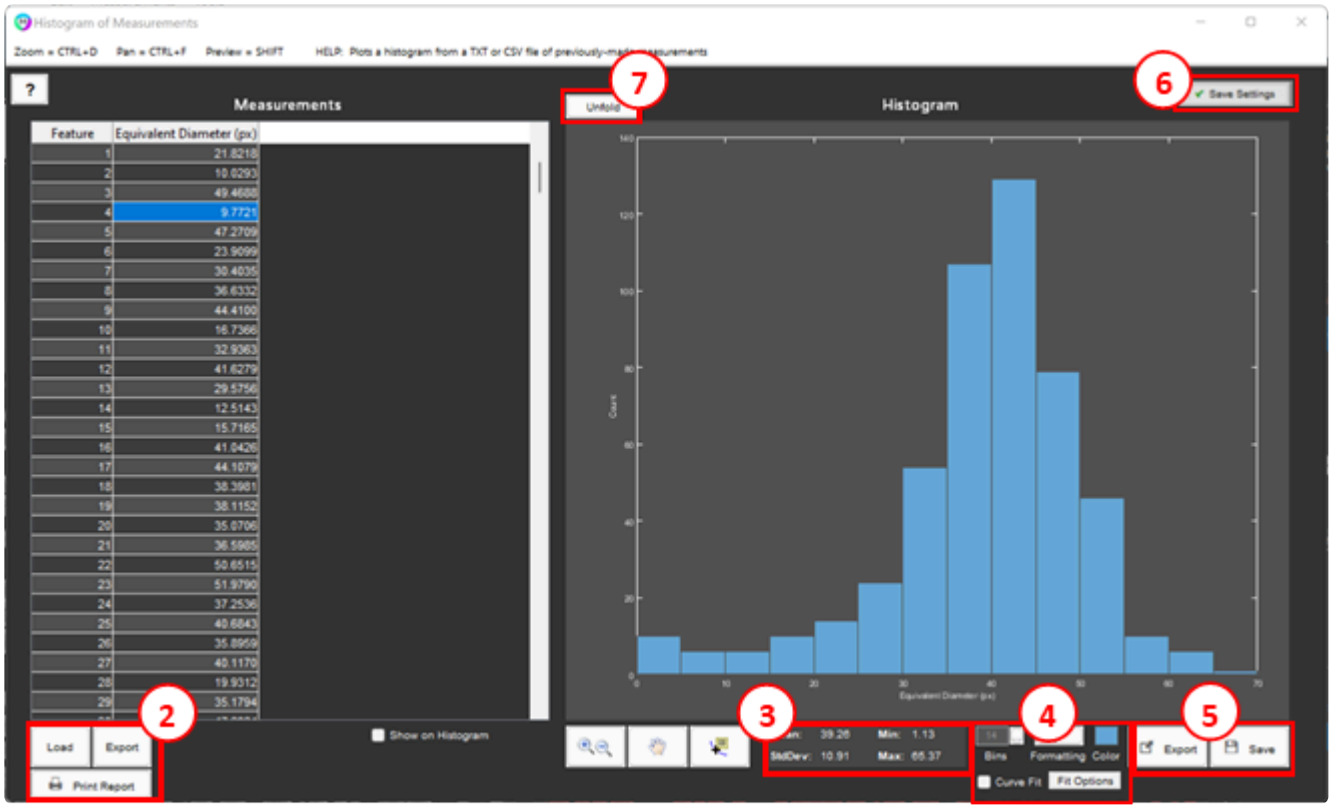
Tools > Histogram of Measurements

Plots a histogram from a .CSV or .TXT file of previously-made measurements. These files can be output from tools such as *Measure Features*, *Color By Measurements*, and *Local Measurements*.



1. Load

- To begin, load previously generated measurements. These must be .TXT or .CSV files with continuous rows and columns.



2. Load, Export

- **Load**: You can start over and create a new histogram image by loading a new feature measurements file.
- **Export**: You can export the existing feature measurements results to a spreadsheet.

3. Measurement Stats

Displays the mean, standard deviation, min, and max of the selected measurements.

> Right-click on these stats and select “Copy” to copy them to the clipboard

4. Histogram Settings

- Determines the number of bins to divide the histogram into. The default value is auto-calculated to provide an effective histogram shape. Typing in a new value sets a manual number of bins. Erase the typed in value anytime to revert back to the auto-calculated value. Click the ellipsis box next to “Bins” to manually set the limits of each bin.
- Formatting changes such as font size
- Bin limits, and axes limits
- Add” Curve Fit”:#curve-fitting to data by picking one of 3 fit types, Normal, Log-normal and Bimodal. Fit Parameters can be viewed by selecting Show Fit Parameters.

5. Export Data, Save Plot

- **Export Data**: Export histogram data to spreadsheet.
- **Save Plot**: Save plot as image.

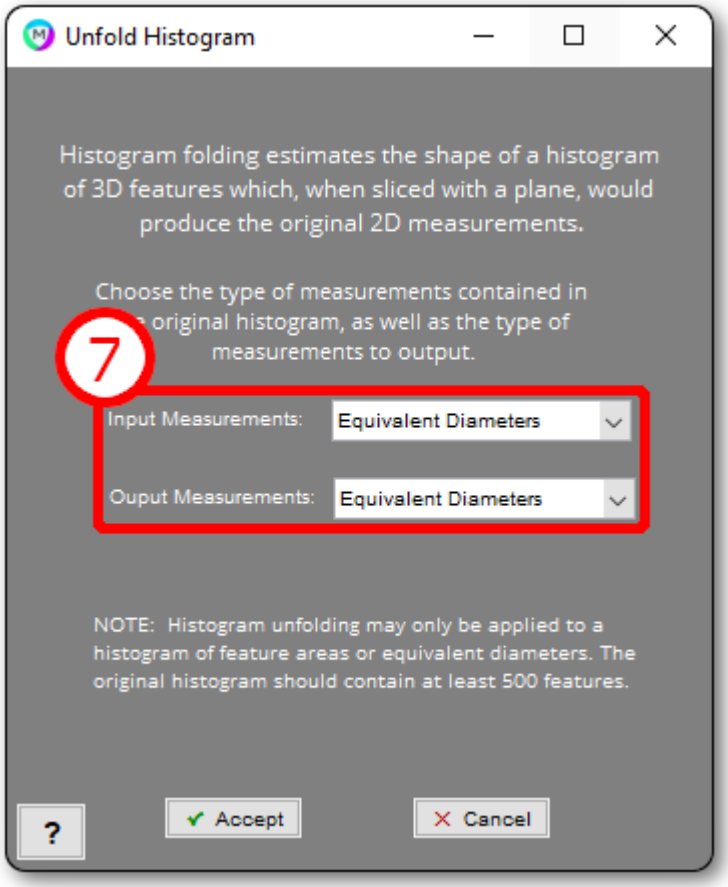
6. Save Settings

If Histogram of Measurements is reached through the play button, then adjustments to display settings, such as selected column, histogram bins, limits, and coloring, will be saved into the recipe. The next time Histogram of Measurements is run on this measurement, these settings will be automatically used.

7. Unfold Histogram

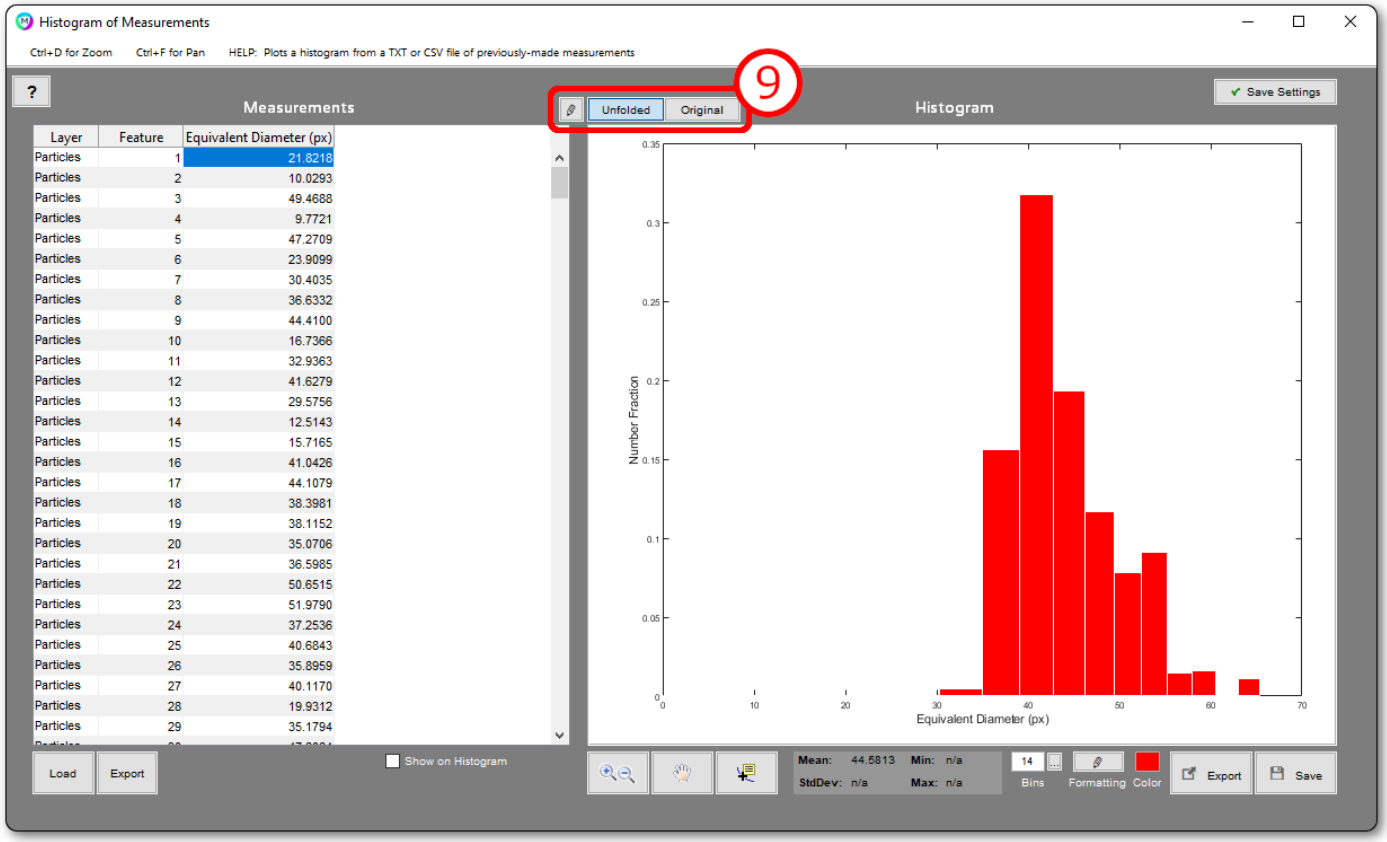
Estimates the shape of a histogram of 3D features which, when sliced with a plane, would produce the original 2D measurements [1]. Launches a window which explains the process and requirements, and allows selection of the input and output measurements.

> Histogram unfolding may only be applied to a histogram of feature areas or equivalent diameters. The original histogram should contain at least 500 features.



8. Input and Output Measurements

- **Input Measurements**: Select the type of measurements to be unfolded. Choose between ***Equivalent Diameters*** or ***Areas***.
- **Output Measurements**: Select the type of measurements to output from the unfolding. Choose between ***Equivalent Diameters*** or ***Volumes***.



9. Original, Unfolded, and Edit

- **Original**: Show original histogram.
- **Unfolded**: Show unfolded histogram.
- **Edit**: Edit histogram unfolding parameters

References

[1] Payton, E. 2012. “Revisiting Sphere Unfolding Relationships for the Stereological Analysis of Segmented Digital Microstructure Images.” Journal of Minerals and Materials Characterization and Engineering 11 (3): 221–242.

Construct Polar Plot

Tools > Construct Polar Plot

Re-plots the rose plot of mean intercept values calculated from a rotating line Intercepts measurement. You must load a “polarPlotData.txt” file output from this type of measurement.

Curve Fitting

Introduction

MIPAR is able to fit a probability distribution curve to a measurement histogram. The supported fit types are: Normal, Log-normal and Bimodal.

Normal Fit

MIPAR uses a Gaussian distribution function to generate a probability curve from the feature measurements.
Equation describing the curve:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

MIPAR reports the mean, standard deviation, and mode using the mu-1 and sig-1 parameters, to display parameters select ‘Show Fit Parameters’

Log-normal Fit

MIPAR uses a Galton distribution function to generate a probability curve from the feature measurements:
Equation describing the curve:

$$\frac{1}{x\sigma\sqrt{2\pi}} \exp\left(-\frac{(\ln(x)-\mu)^2}{2\sigma^2}\right)$$

MIPAR reports the mean, standard deviation, and mode using the mu-1 and sig-1 parameters, to display parameters select ‘Show Fit Parameters’.

Note that mu-1 and sig-1 are the mean and standard deviation of logs and therefore describe the curve and not the mean and standard deviation in the original measurement units. To calculate the log normal fit peak (mode) use this equation:

$$\exp(\mu - \sigma^2)$$

Bimodal Fit

MIPAR uses a Gaussian mixture model with two peaks to generate a fit from the feature measurements. MIPAR reports the mean, standard deviation, mode, and mixing proportion of each component using the mu-1,2; sig-1,2; mix-1,2 parameters, to display parameters select ‘Show Fit Parameters’.

Measurement at Percentile

MIPAR reports the inverse cumulative distribution function values of the Normal and Log-normal fits at probabilities 10%, 25%, 75%, 90%, noted as D10, D25, D75 and D90.

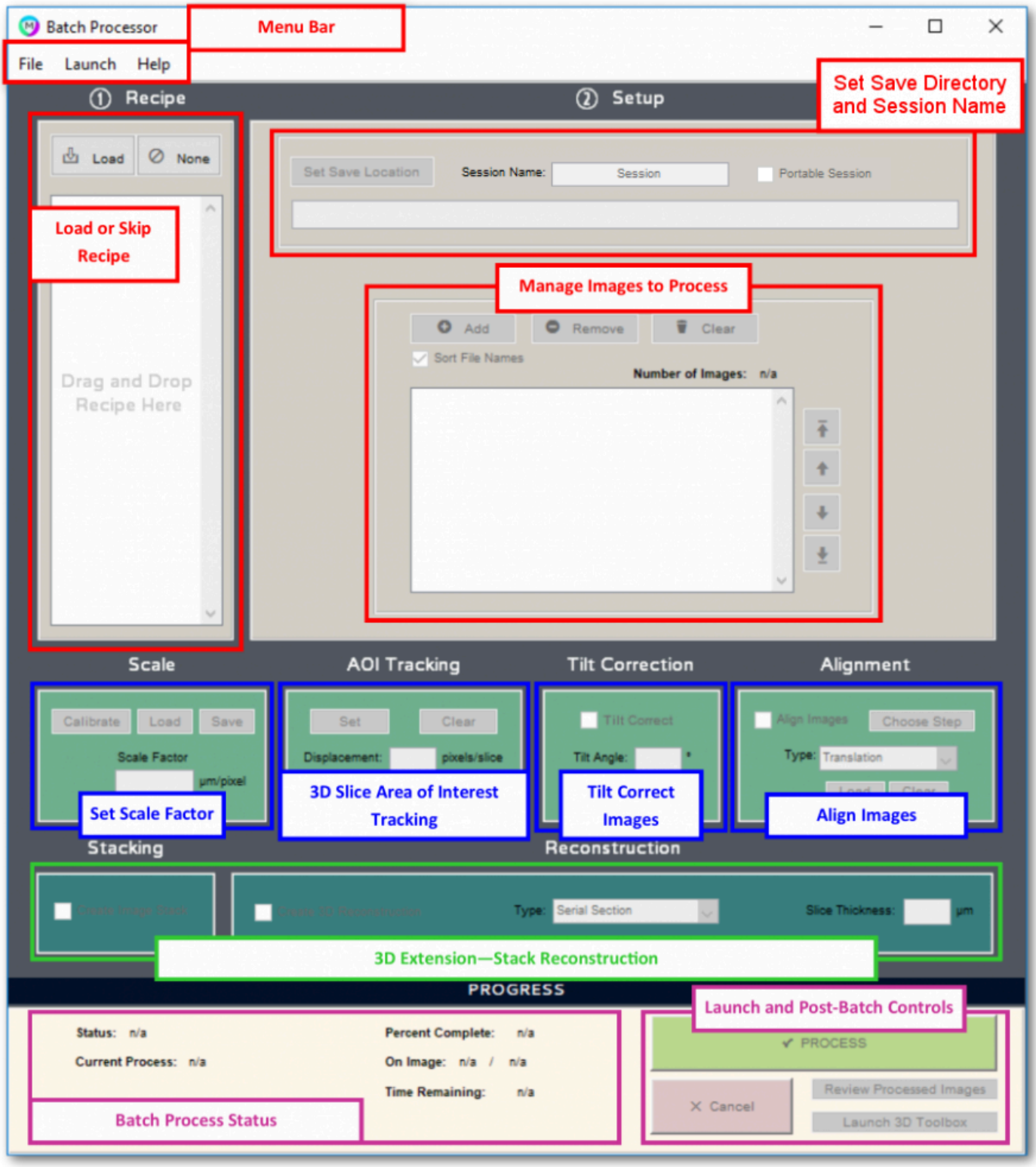
Batch Processor

The Batch Processor is used for batch applying a Recipe to multiple images. A batch process will save the last Recipe step (or each Layer step) for each image, along with a spreadsheet of any global or feature measurements that were set in the Recipe.

Topics

Layout

Below is labeled screenshot of the Batch Processor which reveals the layout of and purpose behind each user interface element.



The Session

Sessions

When running a MIPAR Batch, Real Time Process, building a data set in the Session Processor or AI Session Processor, MIPAR will organize the data into a session.

- In the Batch Processor and Real Time Processor MIPAR automatically collects the detected binary images into folders and matches them up with the reference images. This session can be opened in the (AI) Session Processor to make manual corrections and regenerate the measurements or start a deep learning training.
- In the (AI) Session Processor MIPAR automatically builds a session when putting together data by loading Reference and B/W Images. These sessions can be saved for later or merged with other sessions to create a master data set.

> ! In order for the session files to work, the original reference images and binary images cannot be moved or removed from their original folders. For easier data organization, see Portable Sessions

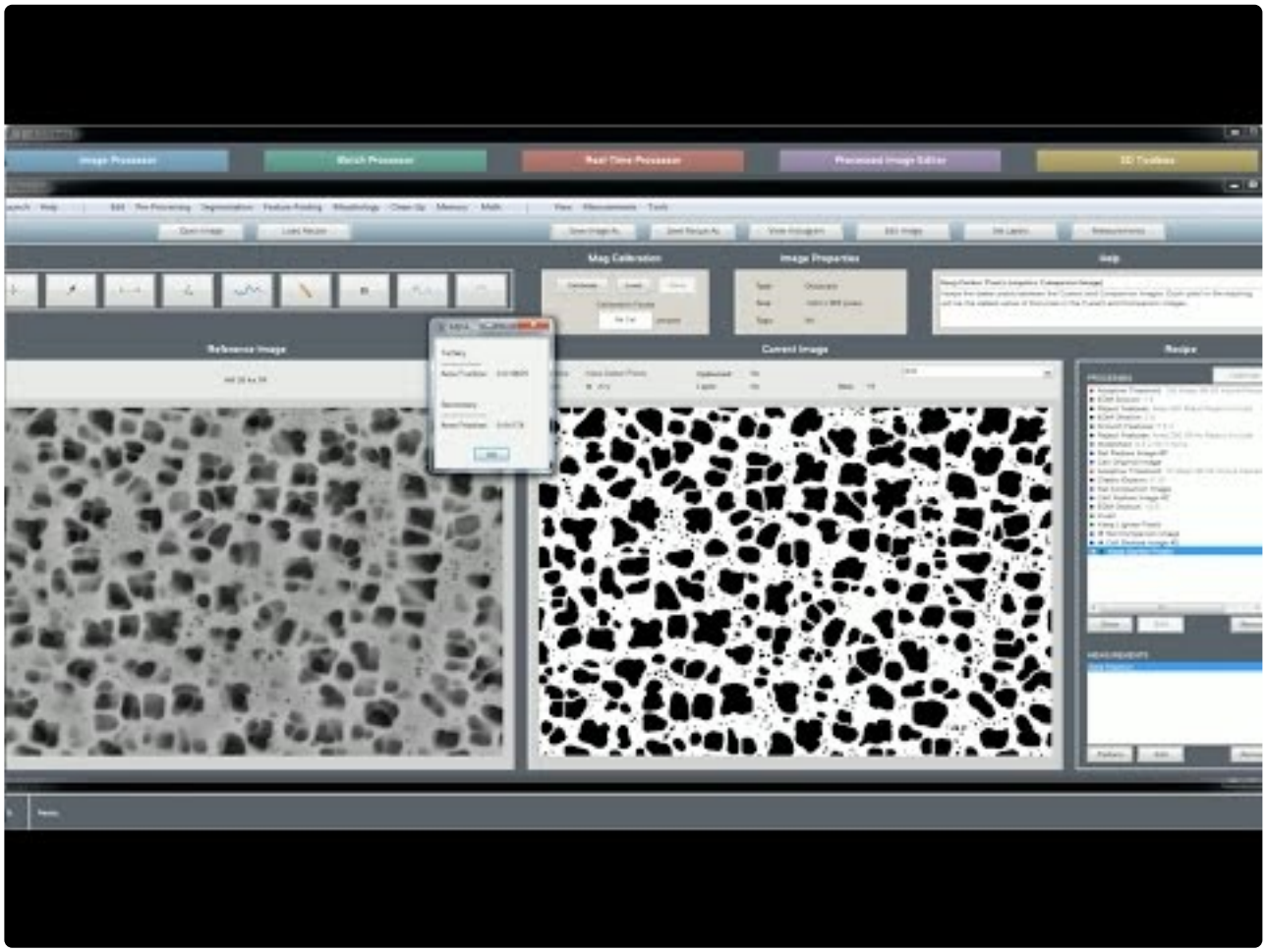
Portable Sessions

MIPAR offers users to save the session as a portable session. This option will organize the reference and binary images together with the session file for easy organization, and data transfer. Keep in mind that this will make a copy of the original reference images which may double the data size.
Even though these sessions are better organized, loading of the session file will fail if the original images were removed or moved from their folders.

Running a Batch

Tutorial

Shows how to setup and run a batch process to segment and measure features from a set images.



https://www.youtube.com/embed/KXbY8vPeroM?rel=0

https://www.youtube.com/embed/KXbY8vPeroM?rel=0

Combine Channels

Using Combine Channels

1. Add images to the batch process that would be combined to a multipage image
2. Check ‘Combine Channels’
3. Specify the number of channels that will be stacked, for example, if every 4 images should be combined to a single multipage image, enter the values 4. The images when sorted must follow the scheme: 1st image is the 1st channel, 2nd image is the 2nd channel, 3rd image is the 3rd channel and so on.

Tutorial Video

Real-Time Processor

The Real-Time Processor is similar to the Batch Processor, but used for applying a Recipe to multiple images, every time a new image is added to a folder being “watched”. Any global or feature measurements added to the recipe will be output into the save directory. This is typically a folder on a network storage drive, such that images may be processed in real-time as they acquired and saved from an imaging instrument.

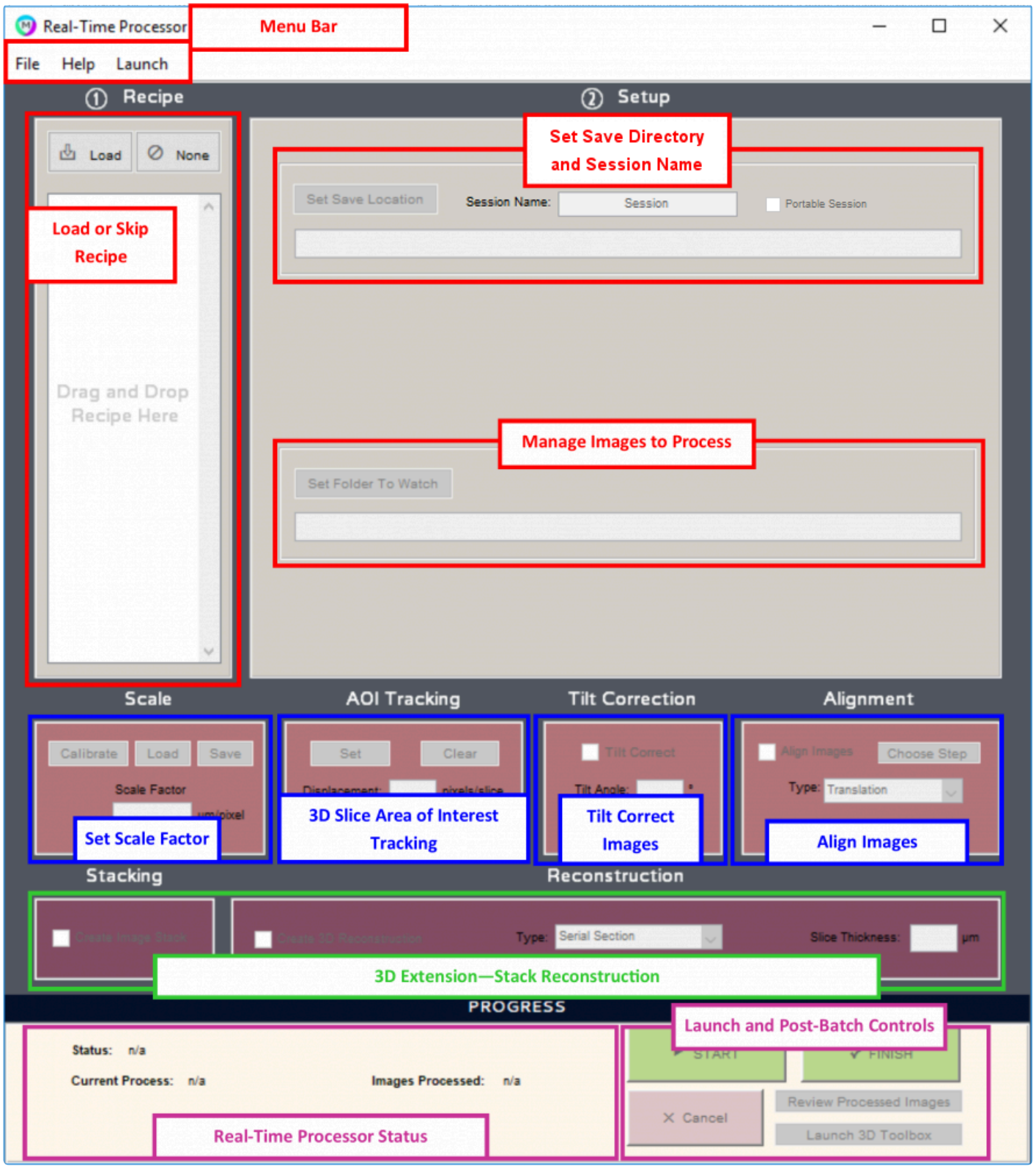
With 3D Extension: Real-Time Processor can output Image Stacks and Reconstructions

Topics

- Layout

Layout

Below is labeled screenshot of the Real-Time Processor which reveals the layout of and purpose behind each user interface element.



Session Processor / AI Session Processor

Application Description

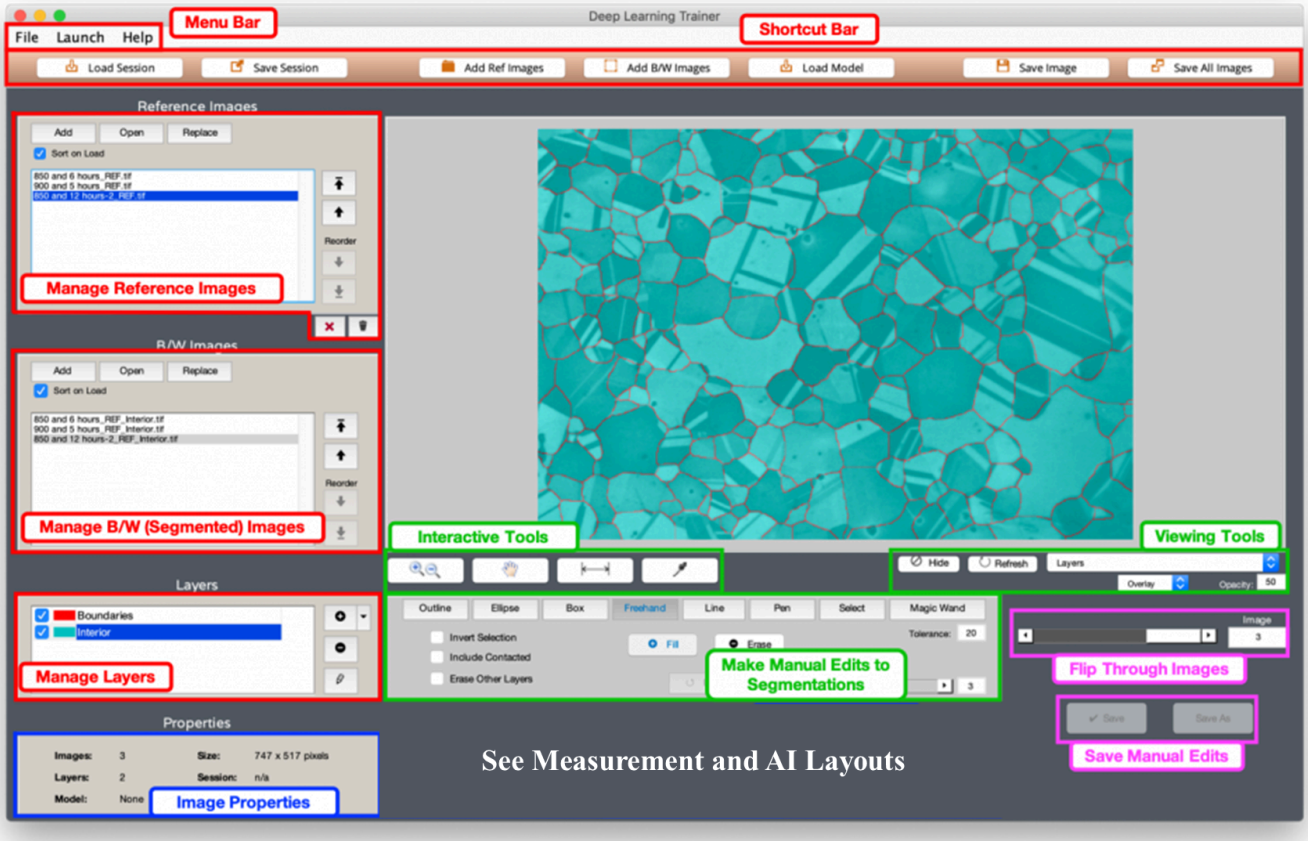
The Session Processor and AI Session Processor is used to review Batch Process Sessions, generate measurements on a data set and train new Deep Learning Models.

Topics

- Layout
- Measurements
- AI Training

General Layout

Below is labeled screenshot of the Post Processor which reveals the layout of and purpose behind each user interface element.



- Measurements Layout
- AI Layout

> See the Keyboard Shortcuts page for shortcuts relevant to the Post Processor.

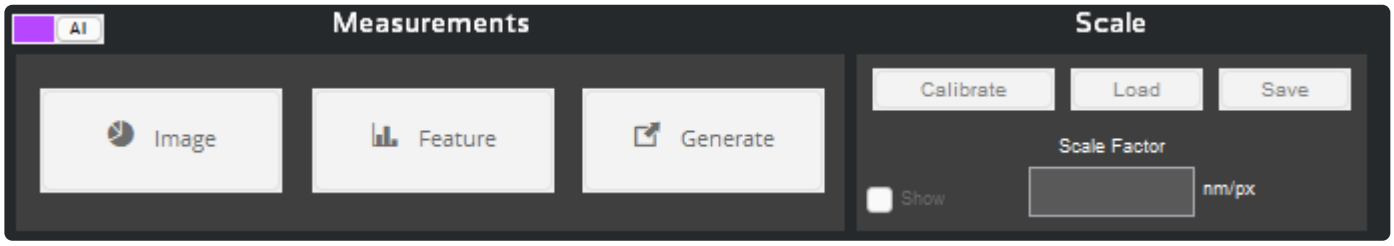
Measurements (formerly Post Processor)

The Session Processor is used for reviewing the results of a batch or real-time process. All saved processed images are loaded and overlaid/outlined, etc. on their respective references. The user can flip through each processed result to assess accuracy. Manual edits can be made to correct errors. Global and feature measurements can be made from all processed images at once.

Topics

Measurement Layout

Measurement Panel Layout



Image

Add global measurements, for example; total layer area, layer feature count, layer area fraction.

Feature

Add feature measurements, for example; area per feature, roundness per feature, nearest neighbor distance per feature. View the results table and histogram, save the results to file or add to the summary table report. (See Generate)

Generate

Generate a summary table for both Image and Feature measurements added.

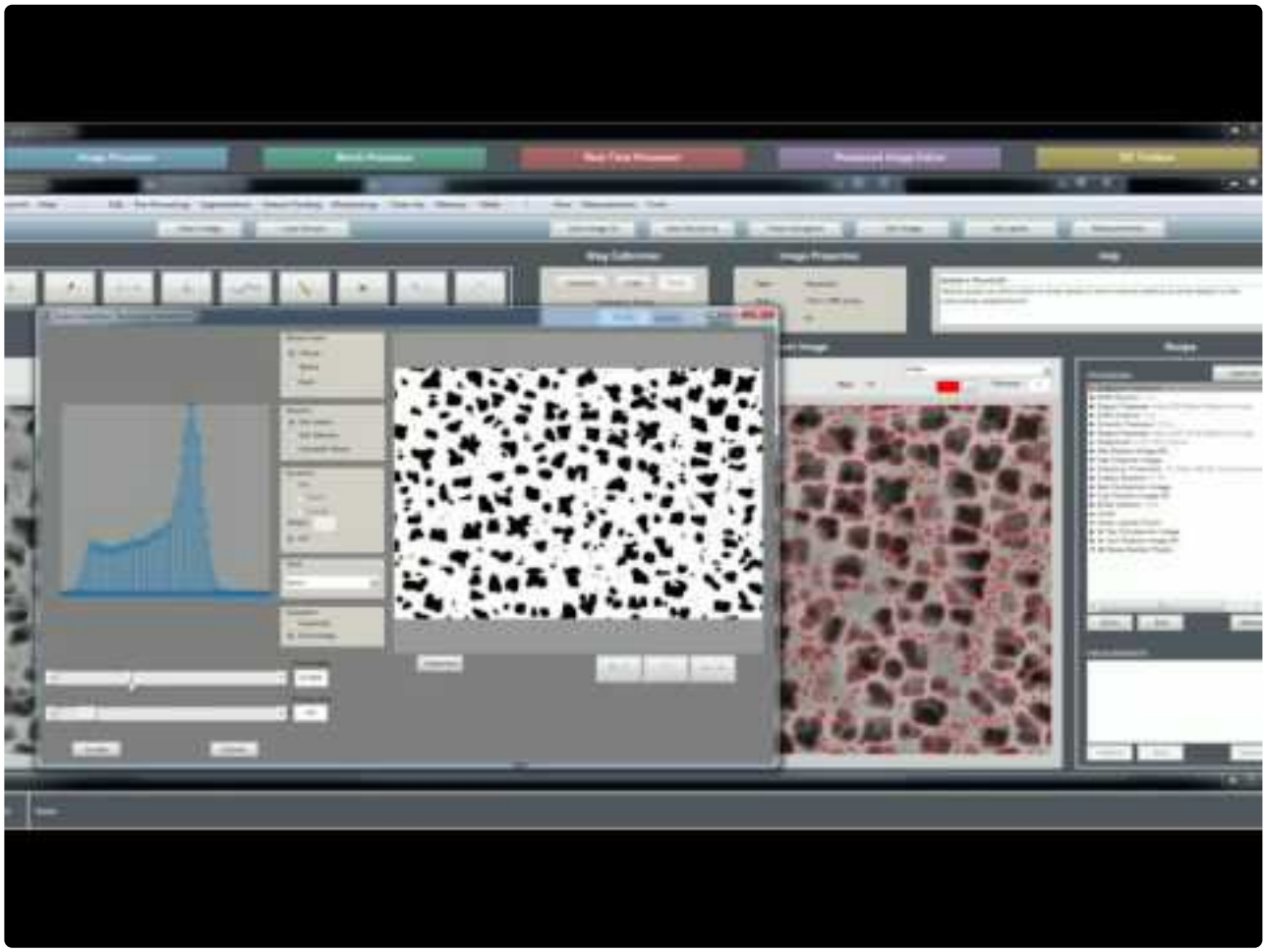
Scale

“ session to convert pixel units (px) to physical units (cm, mm, in, etc)

Reviewing a Batch

Tutorial

Shows how to review and edit multiple images after they have been batch processed to segment features of interest.



https://www.youtube.com/embed/whP1423oY1k?rel=0

https://www.youtube.com/embed/whP1423oY1k?rel=0

Manual Editing

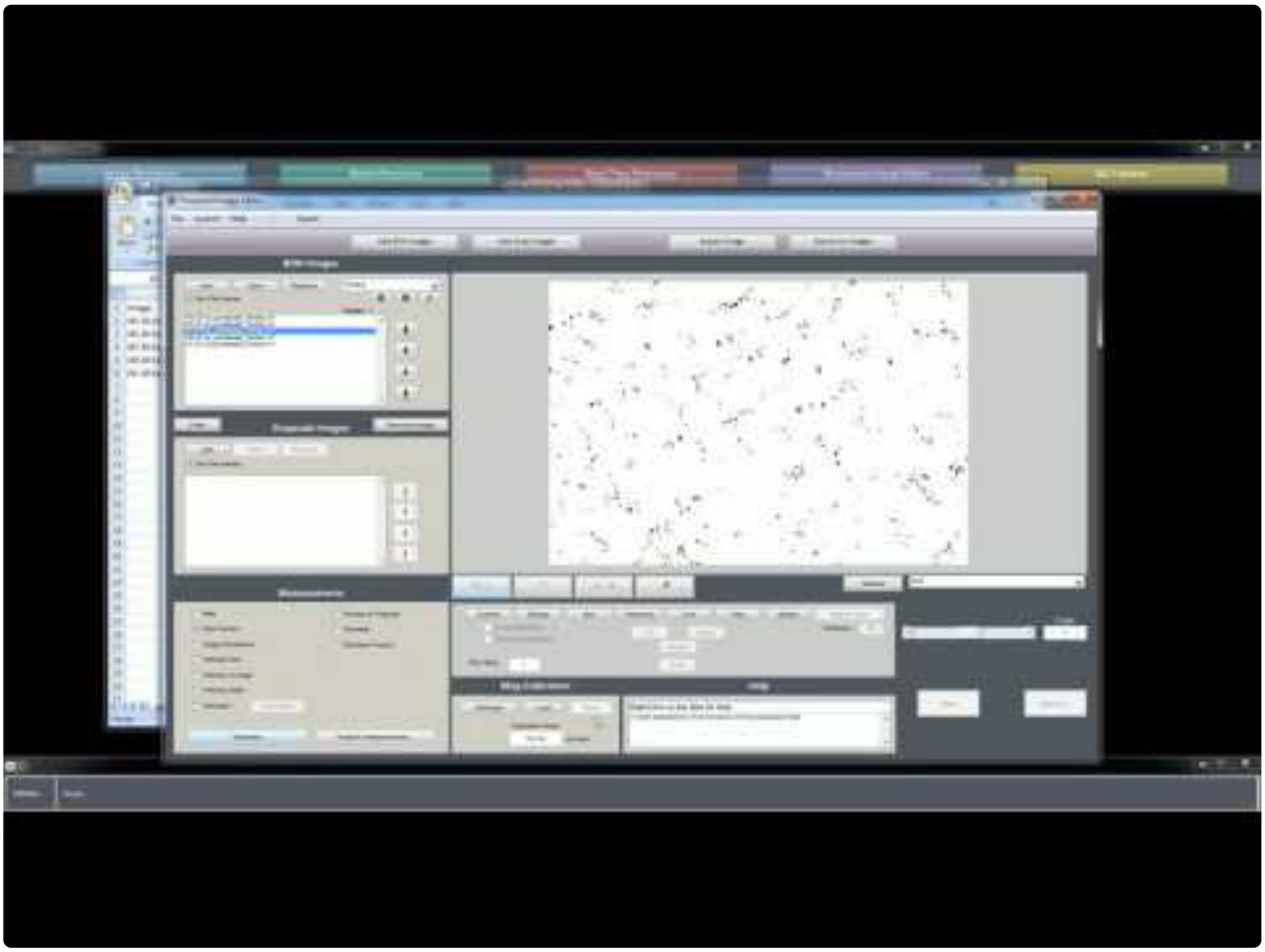
For a list of available manual editing tools and their descriptions, please click the links the below:

- Manual Editing Tools *(same tools in this editor apply to the Session Processor)*

Bulk Measurements

Tutorial

Shows how to review and edit multiple images after they have been batch processed to segment features of interest.



https://www.youtube.com/embed/KyzyMeVz0bM?rel=0

https://www.youtube.com/embed/KyzyMeVz0bM?rel=0

Measurements

For lists of available measurements and their descriptions, please click the links below:

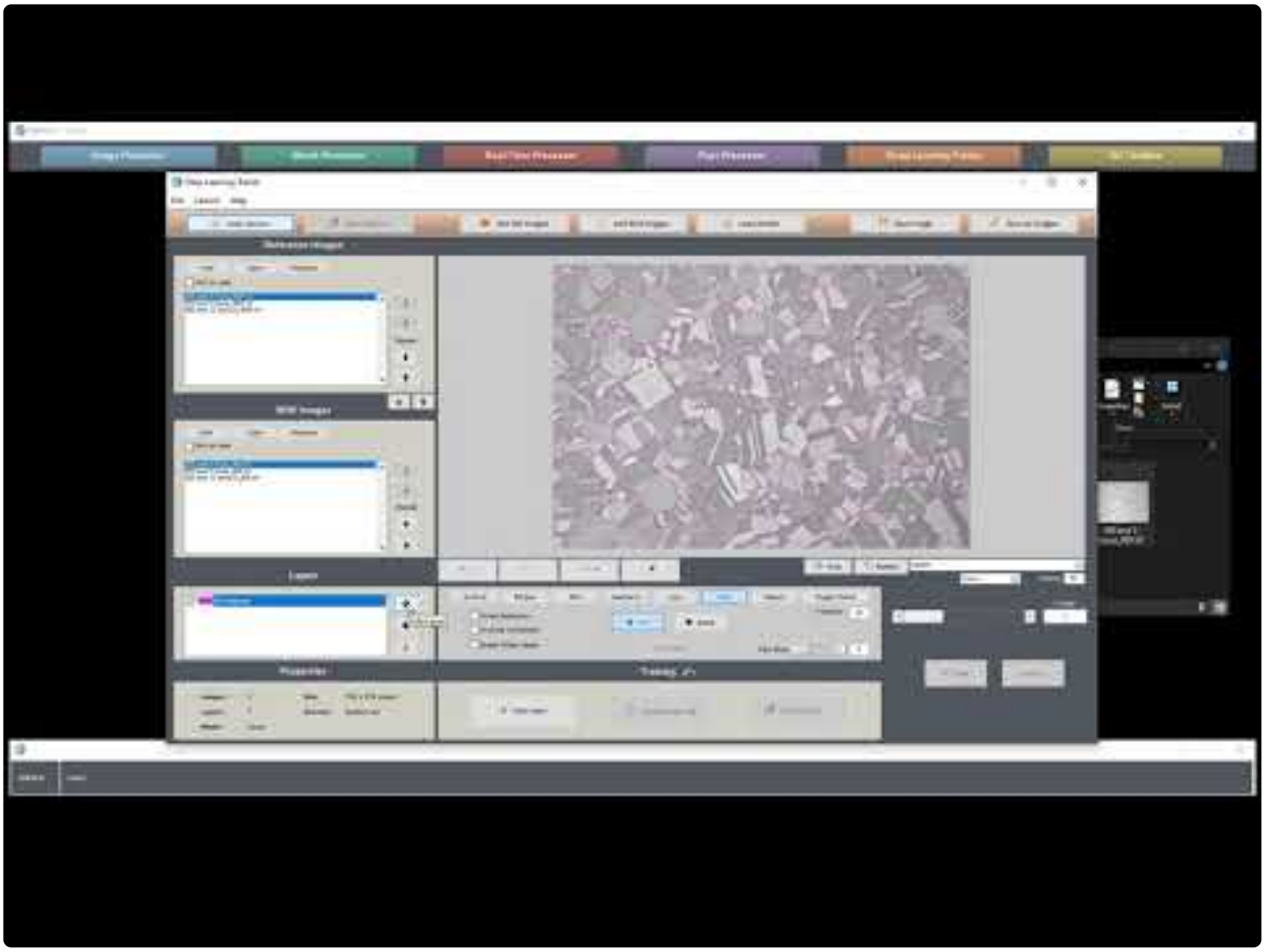
- Global Measurements
- Feature Measurements

AI (formerly Deep Learning Trainer)

Requires Deep Learning Extension

The Deep Learning Trainer is used to train deep learning models, which are capable of automating some of the most challenging feature detection problems across many applications.

Below is a short video demonstrating the use of the Deep Learning Trainer:



https://www.youtube.com/embed/ACvg6LAsqRY?rel=0

https://www.youtube.com/embed/ACvg6LAsqRY?rel=0

These models may be applied as part of a Recipe by adding an *Apply Model* step in the Image Processor.

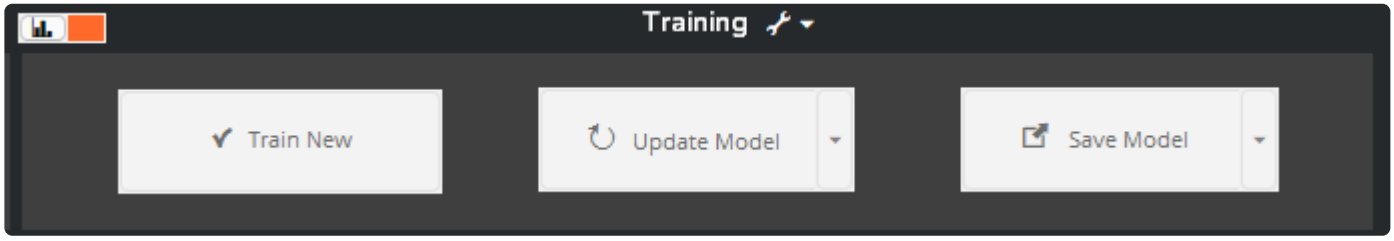
Topics

- Updating
- Applying

AI Layout

Below is labeled screenshot of the AI Session Trainer which reveals the layout of and purpose behind each user interface element.

Layout



Train New

Train new deep learning model using the reference and BW images and the Training Settings

Update Model

Update deep learning model with new images/layers in the session.

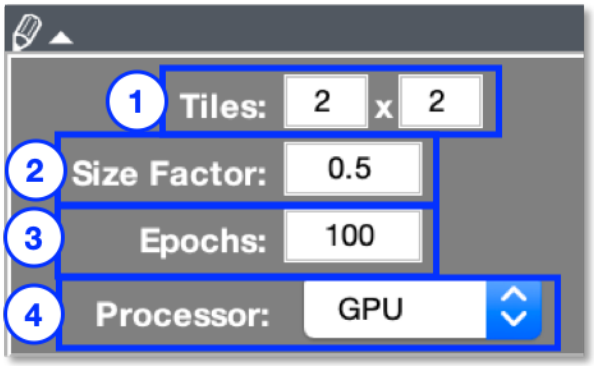
Save/Apply Model

Save the trained model/apply the model in the image processor to the reference image in focus.

> See Keyboard Shortcuts for shortcuts relevant to the Deep Learning Trainer.

> Session files are cross-compatible between the Deep Learning Trainer and Post Processor.

Training Settings



1. Tiles

Grid spacing to split each image into tiles prior to training. This reduces memory load on the GPU and provides more data to the training algorithm, which tends to produce better results. *(Example: Tiles = 3×2 splits each image into 6 sub-images)*

2. Size Factor

Resize factor for training images (0-1). This reduces GPU memory load and significantly speeds up training times. A factor down to 0.5 can often be used and still provide adequate precision.

3. Epochs

Number of epochs to run training for. The below table offers some recommendations based on your amount of training data:

Number of Training Sub-images	Epochs
< 200	500-700
200-500	300-500
500-1000	100-300
> 1000	50-100

4. Processor

- **GPU:** Train model on single GPU (strongly recommended — see Deep Learning System Requirements for more information). Active GPU is set in “GPU Computation” panel in *File* > *Preferences*.
- **CPU:** Train model on the CPU
- **Multi-GPU:** Train model on multiple GPUs on your local system (only available when more than 1 GPU is detected)

AI Training System Requirements

Training

GPU

A GPU is strongly recommended for training deep learning models. Training performance on a GPU can be up to 20x faster than on a CPU, and can shorten training time from weeks to hours.

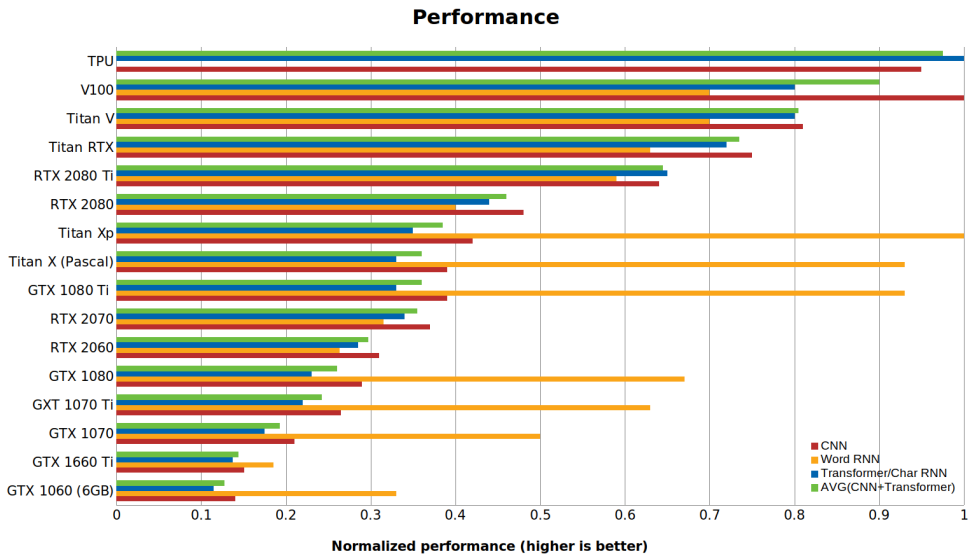
> GPU training requires an NVIDIA GPU with **compute capability 3.5 or higher** (Kepler, Maxwell, Pascal, Volta, Turing, or Ampere architecture), and your driver must support **CUDA Toolkit 11 or higher**. We recommend you update your NVIDIA driver to the latest version here. Learn more about CUDA here.

Below is a list of recommended GPU options:

	Basic	Mid-Range	Performance	High-End
GeForce GTX 1060	x			
GeForce GTX 1070	x			
GeForce GTX 1080		x		
Quadro P4000		x		
GeForce RTX 2060		x		
GeForce RTX 3060		x		
GeForce RTX 2070			x	
GeForce RTX 3070			x	
Quadro RTX 4000			x	
GeForce GTX 1080 Ti			x	
Titan Xp			x	
GeForce RTX 2080			x	
GeForce RTX 3080			x	
GeForce RTX 2080 Ti				x
GeForce RTX 3080 Ti				x
Quadro RTX 6000				x

Titan RTX				x
Titan V				x
Tesla V100				x

Below is chart of benchmark comparisons between various cards. The red bar is most relevant to deep learning training performance.



Source: https://timdettmers.com/2019/04/03/which-gpu-for-deep-learning/

Applying

GPU

A GPU is also recommended for applying deep learning models. Application performance on a GPU can be up to 20x faster than on a CPU.

CPU

Since applying often takes seconds, compared to hours for training, a CPU can more reasonably be used if necessary. Please refer to the general System Requirements for more information on CPU requirements and recommendations.

Tracing

There are several approaches to tracing features to setup a deep learning training. Below are examples of three recommended approaches. Tracing can be done directly in the Deep Learning Trainer (fully manual, or semi-automated), in the Image Processor, or in an external application.

> **Tip:** The quality of model training will depend on tracing quality, so take care to trace features as accurately as possible. Tracing all features of interest is recommended, but not required.

> **Tip:** Training with more layers tends to produce better results. For example, consider using 3 or 4 layers, when you might have only used 2. *(Example: Detecting faint pores in the presence of dark artifacts. Use at least 3 layers: pores, artifacts, and background.)*

Tracing in AI Session Processor

Fully Manual

This example shows a fully manual approach to tracing features directly in the AI Session Processor.

Downloads

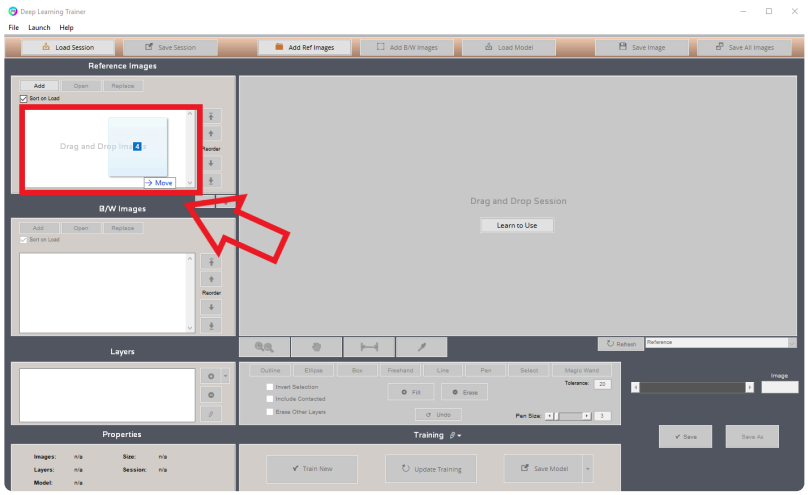
You may download the sample reference images to use as learning tools and follow this example.

Reference Images

Load References

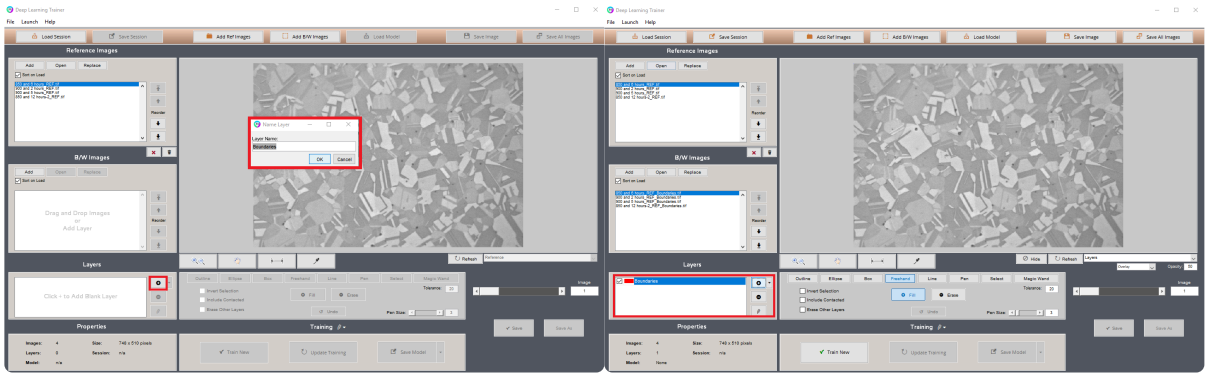
Begin by launching **AI Session Processor** from the Launch Bar.

Drag and drop your reference images.



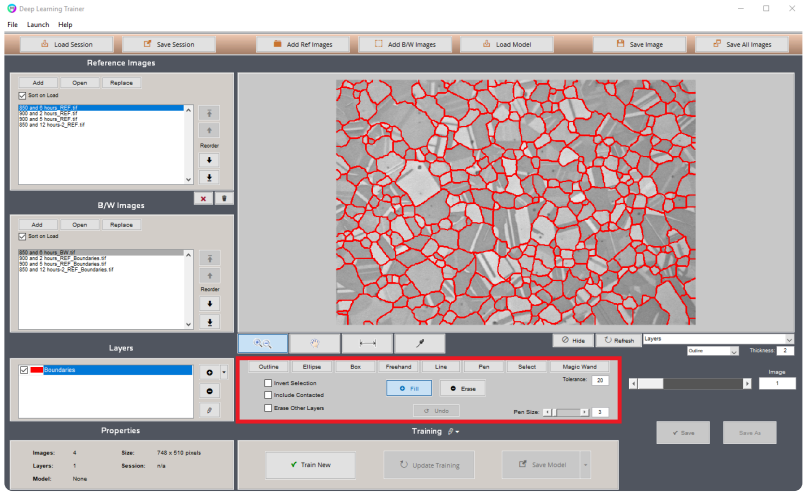
Add Layers

Click **+** in the Layers Panel to add a blank layer. Set the name and click “OK”. You may add additional Layers at any time using the **+** button. Click the Layer name in the Layers Table to set the active Layer for tracing.



Trace Features

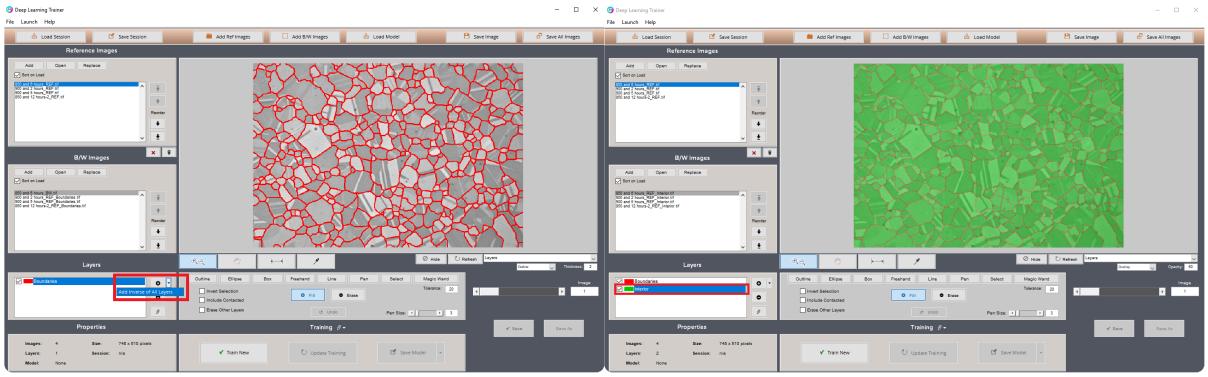
Use manual editing tools to trace features of interest.



> **Tip:** The quality of model training will depend on tracing quality, so take care to trace features as accurately as possible. Tracing all features of interest is recommended, but not required. Pixels which are not assigned to any Layer will be ignored during training.

Click **Save** and advance to the next image to continue tracing.

After all features have been traced in each image, you may use the **dropdown arrow** next to the **+** followed by **Add Inverse of All Layers** to add a background Layer if needed. Name the Layer and click “OK”. You’ll usually choose “No” when asked if you want to erode after inversion. This will add all undefined pixels as another Layer.



> **Tip:** Training requires at least 2 Layers. Only use the above inversion method if all features in Layer 1 are traced. If they are not, click **+** to add the second Layer, and trace a few areas of the background.

Semi-automated

This example shows a semi-automated approach to tracing features in the AI Session Processor. This approach is useful if you have a Recipe which does a reasonable job of detecting features of interest, but leaves room for improvement

Download example Recipe and image

Batch Apply

Launch the **Batch Processor**.

Load Recipe and drag and drop images to batch process.

Click **Process**.

Clean Up

When batch is finished, launch **AI Session Processor**.

Drag and drop the Session file which was saved to the output directory of the batch process.

Use manual edit tools to correct any detection errors in all Layers of the first image.

Click **Save** and advance to the next image to continue correcting. Repeat until all images are corrected.

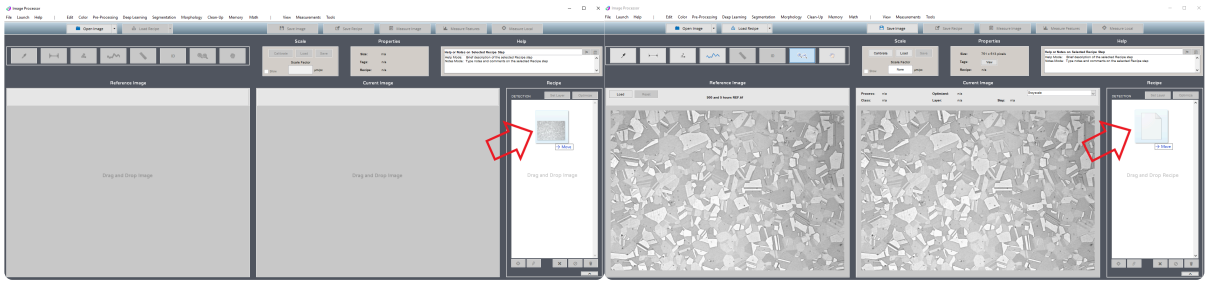
Tracing in Image Processor

This example shows a Recipe-assisted approach to tracing features in the Image Processor. This approach is most useful for tracing grain boundaries, or other similar features.

Load Image and Recipe

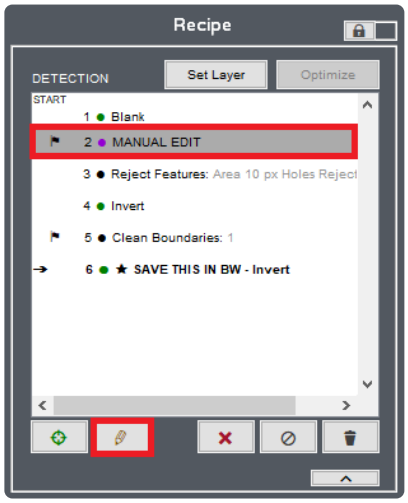
Download “trace-grains” recipe. This can be used as a “starter-kit” for any Recipe-assisted tracing of grain boundaries (or similar features).

Drag and drop your first image to open. Drag and drop “trace-grains.rcp” to load.

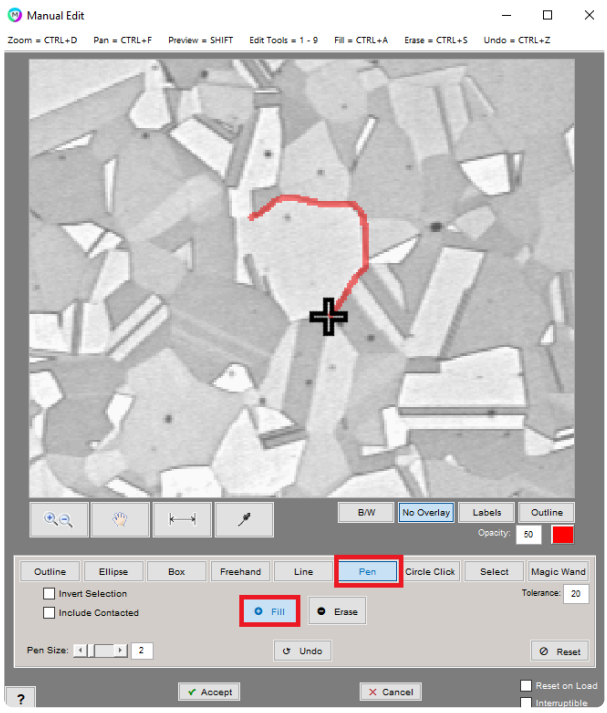


Trace Features

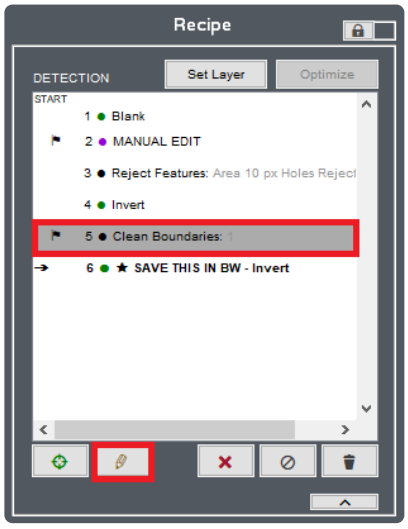
Select **Manual Edit** (first flagged step) and click **Edit** (pencil icon beneath Recipe).



Use the manual edit tools in this window to roughly trace boundaries. It is recommended that you click **Accept**, save the Recipe, and re-open the editor every so often to preserve your work.



If needed, select **Clean Boundaries** and click **Edit** to adjust the “Thickness” setting. This controls final tracing thickness.



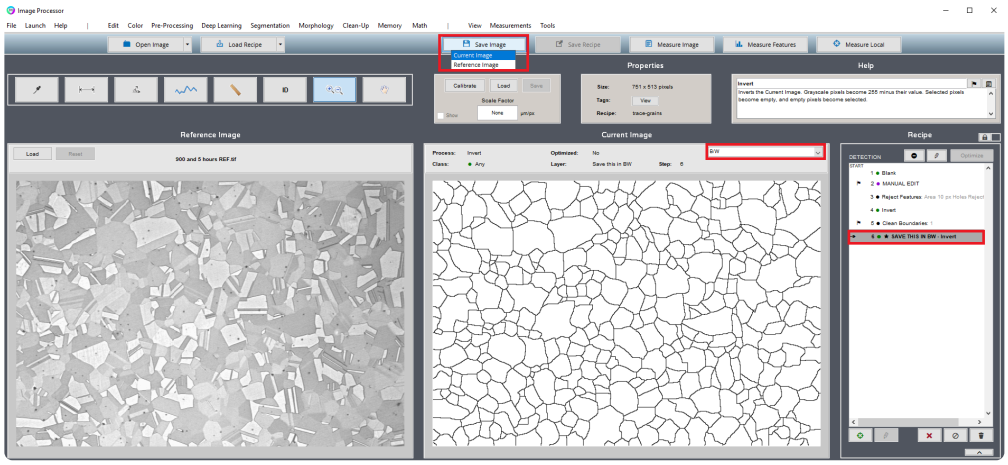
> **Tip:** Use as thin a line as possible, while still covering important pixel information at the boundaries. A thickness of 1-3 usually works best.

Save Results

When you have finished tracing on this image, double-click the last Recipe step.

Switch viewing mode to “B/W”.

Click Save Image > **Current Image** and save the image with a unique name *(e.g., add _BW to the end of the file)*.



Repeat this example for each image you have to trace.

> **Tip:** Save a separate copy of this Recipe after you trace each image. This will allow you to recall each image's full edit history and make changes later.

Tracing in External Application

There are several applications available for manual image tracing on desktop and mobile operating systems. Below are just a few of the options for Windows and Mac platforms:

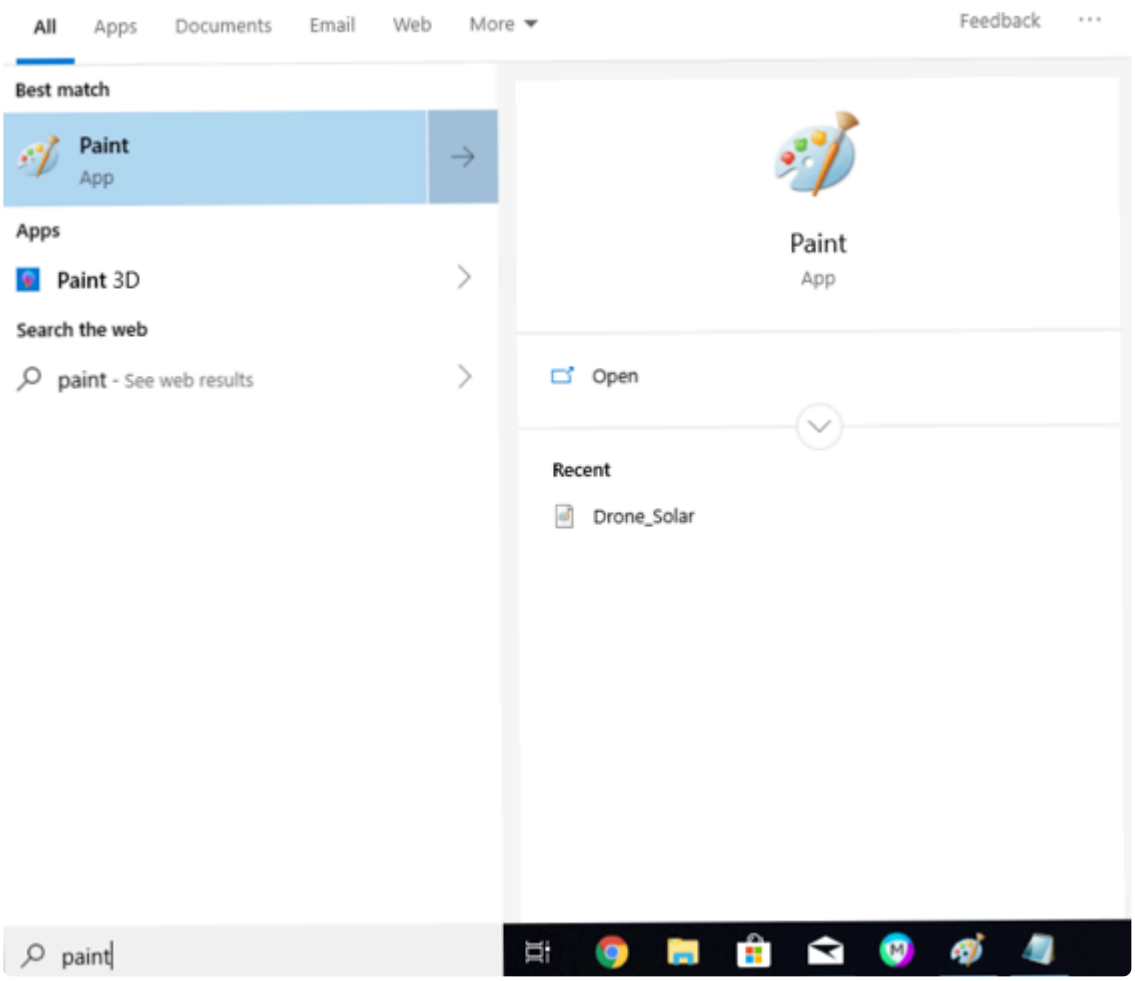
	Windows	Mac
Adobe Photoshop	x	x
Gimp	x	x
Microsoft Paint	x	
Preview		x

Trace in Paint (Windows)

This example shows an approach to tracing features in Microsoft Paint, such that they can then be imported into MIPAR for deep learning training.

Load Image

Open ‘Paint’ application.

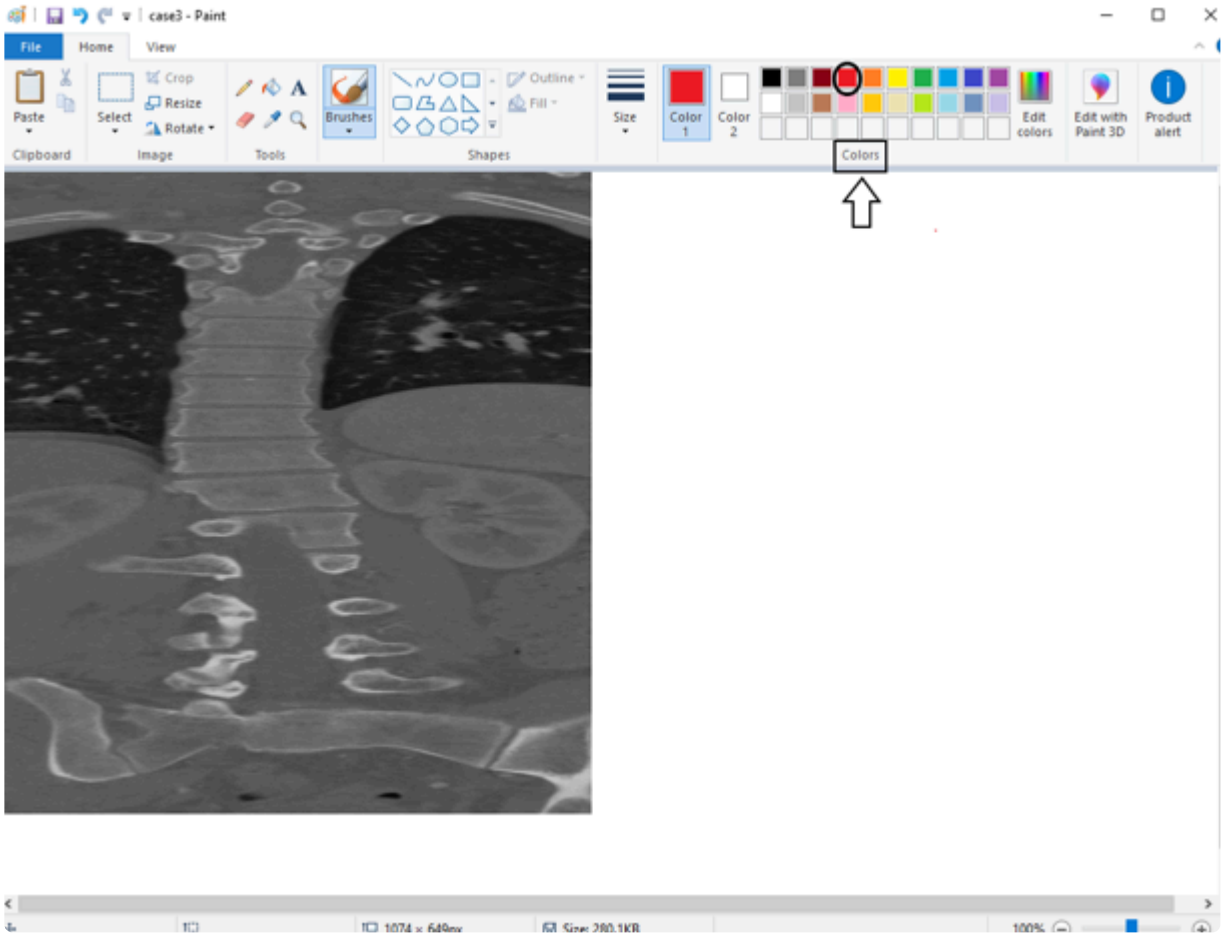


Drag and drop image into ‘Paint’.

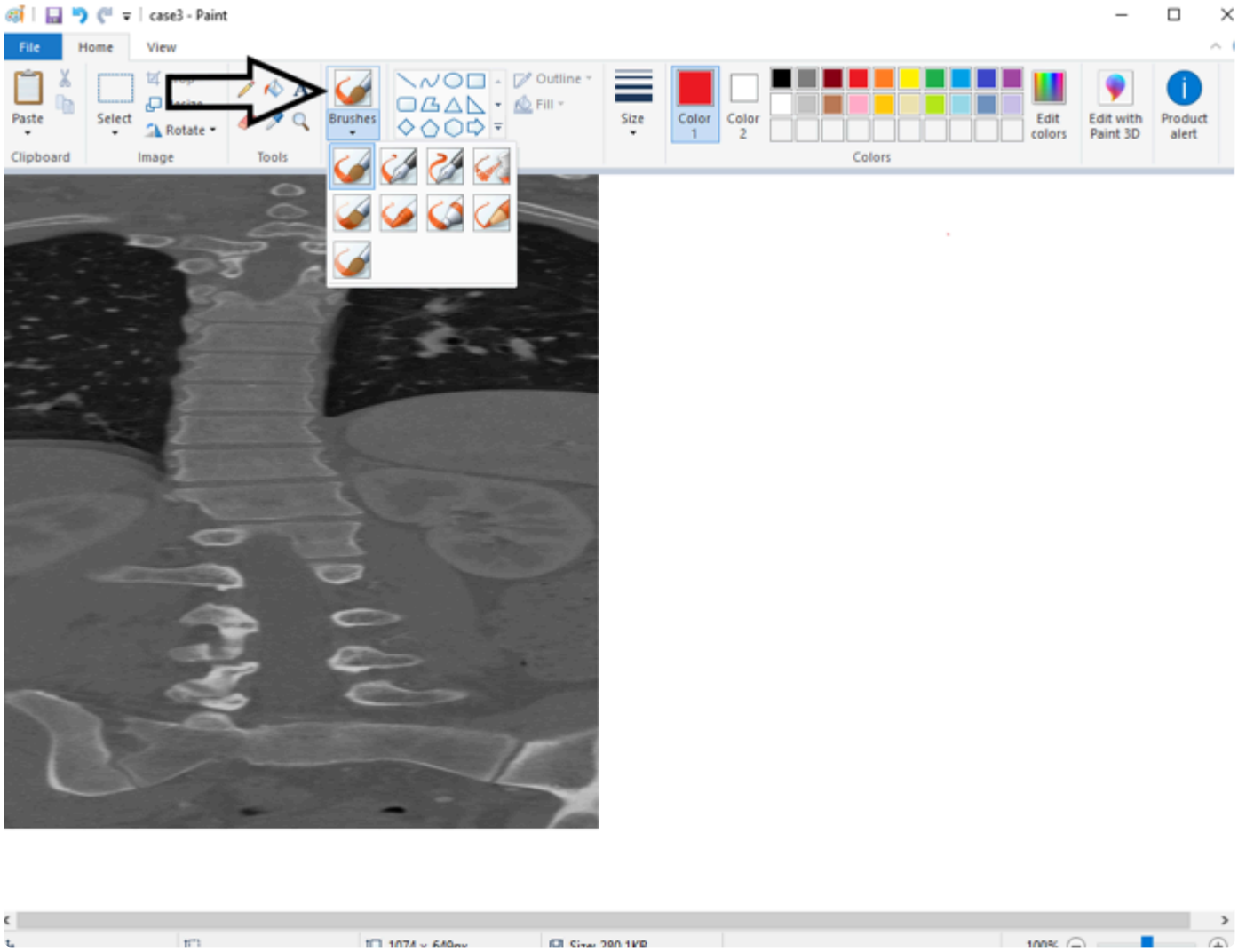
Trace Features

Select color ‘Red’ (Class 1), Blue (Class 2), Yellow (Class 3), Green (Class 4).

> **Tip:** Each class should be a different item of interest, the more classes produce significantly better results i.e. background, negative classes (things we don’t want).

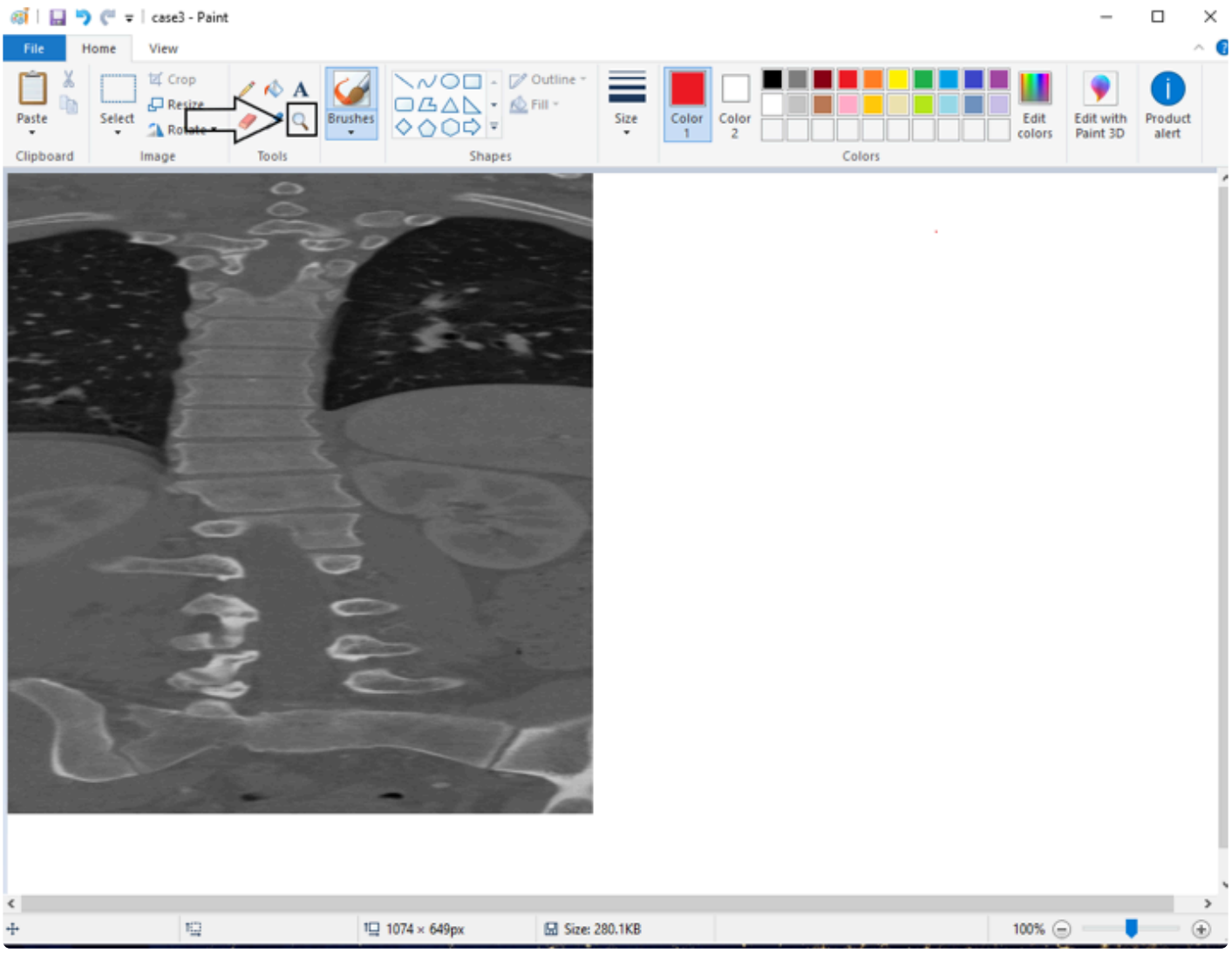


Select any pen from ‘Brushes’ drop down menu.



Left click and hold the mouse to draw/trace areas of interest .

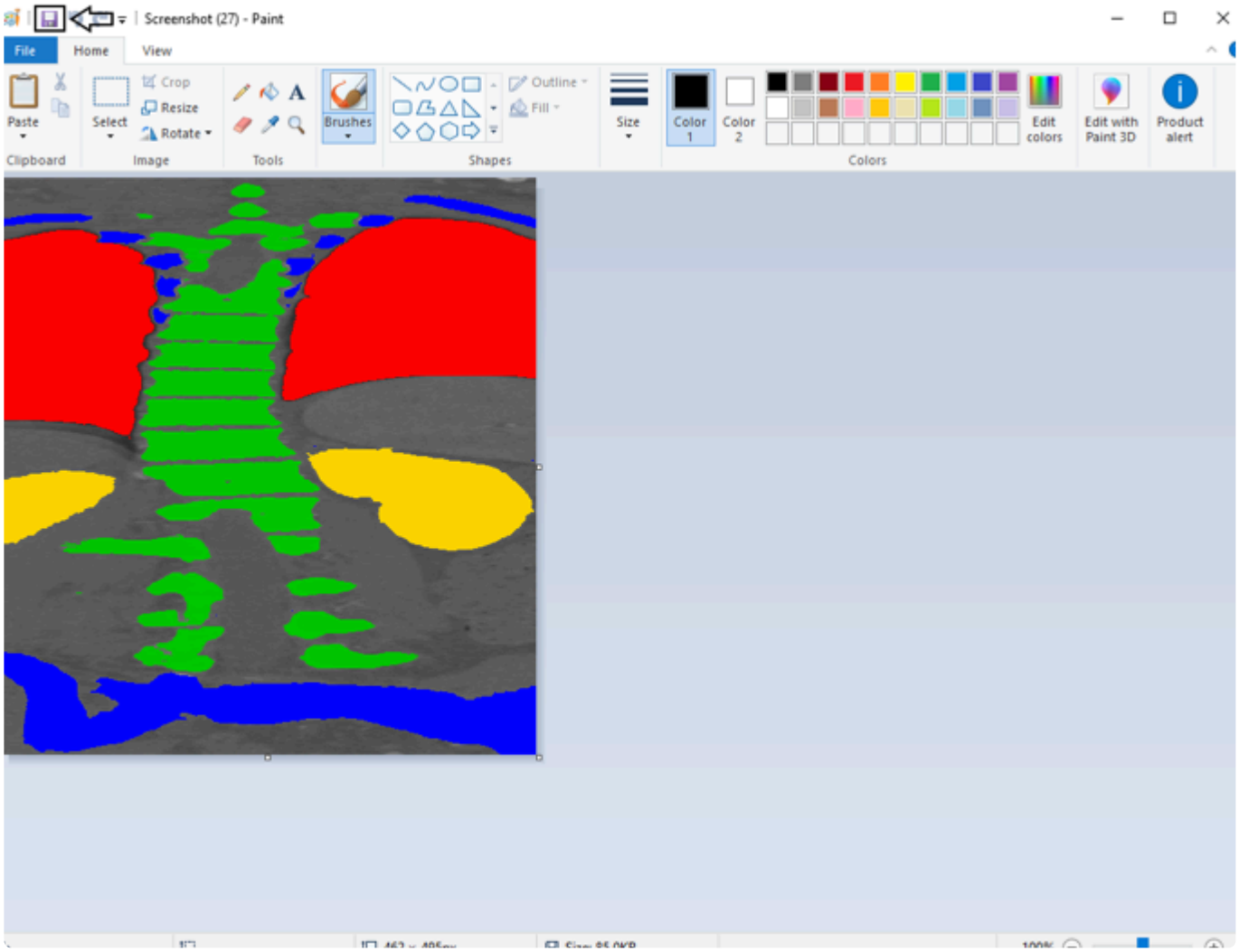
Use magnify glass to zoom in on areas of interest If you make a mistake and would like to delete it hold ‘Ctrl + z’ to remove.



Save Results

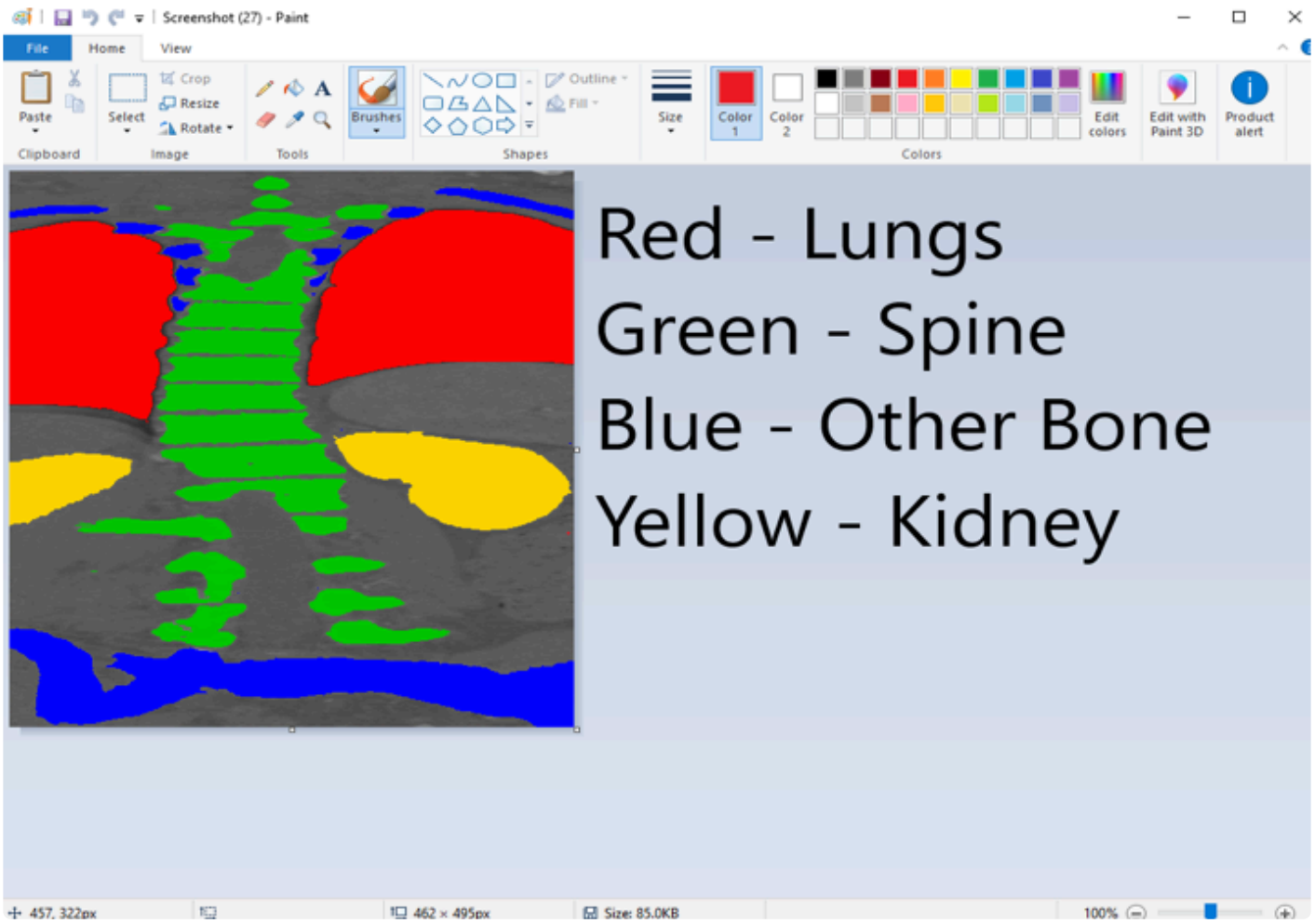
Once finished tracing, press the floppy disk icon in bar to ‘save’ image .

> **Tip:** Save in either TIF or PNG format to avoid blurring at the edges of your tracings.



Example Tracings

We want to identify the spine, but also selected three other classes in order to give the algorithm more data to learn from. Using these negative classes gives the software more information to know exactly what makes your desired features unique.

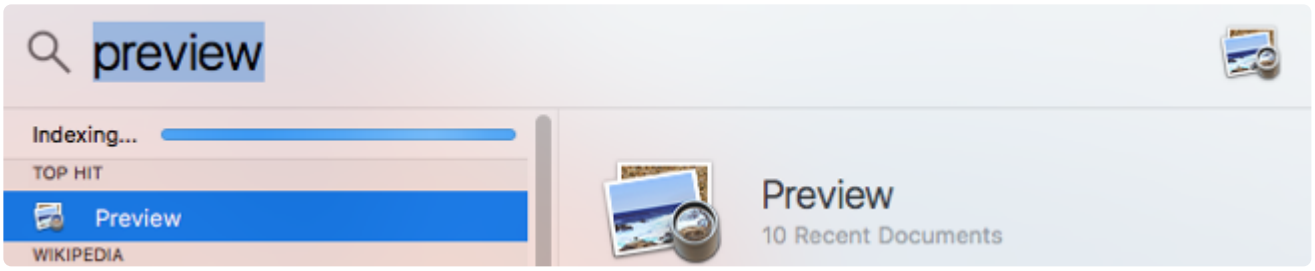


Trace in Preview (Mac)

This example shows an approach to tracing features in Preview on Mac systems, such that they can then be imported into MIPAR for deep learning training.

Load Image

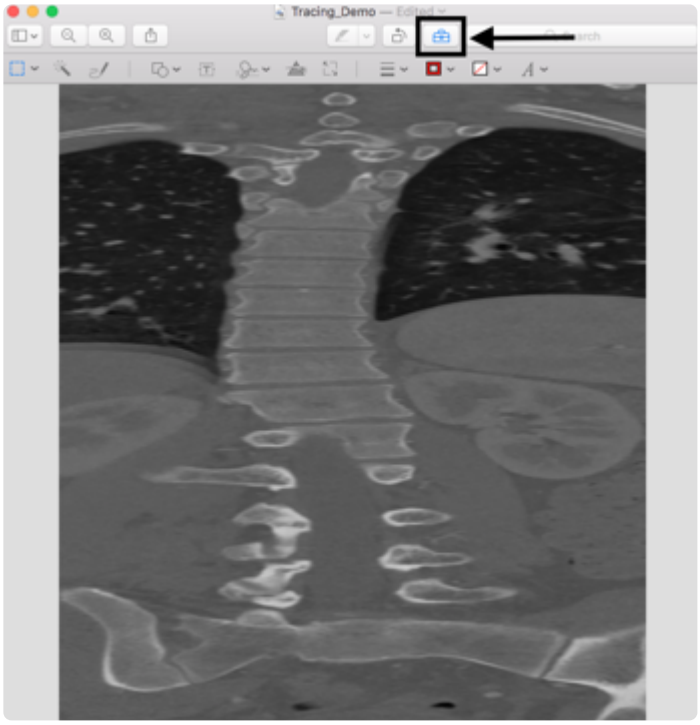
Open ‘Preview’ application.



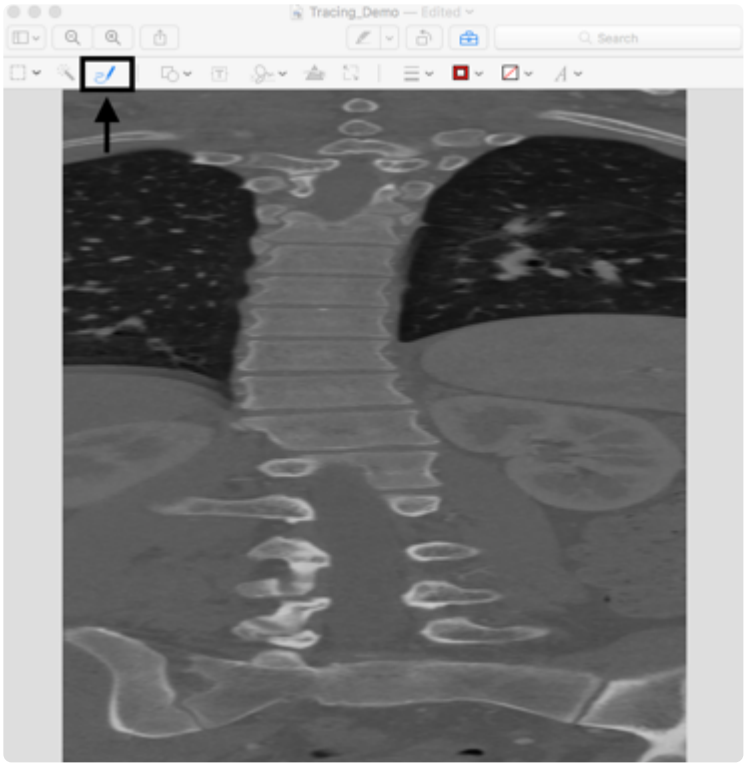
Select your image file from your computer drive

Trace Features

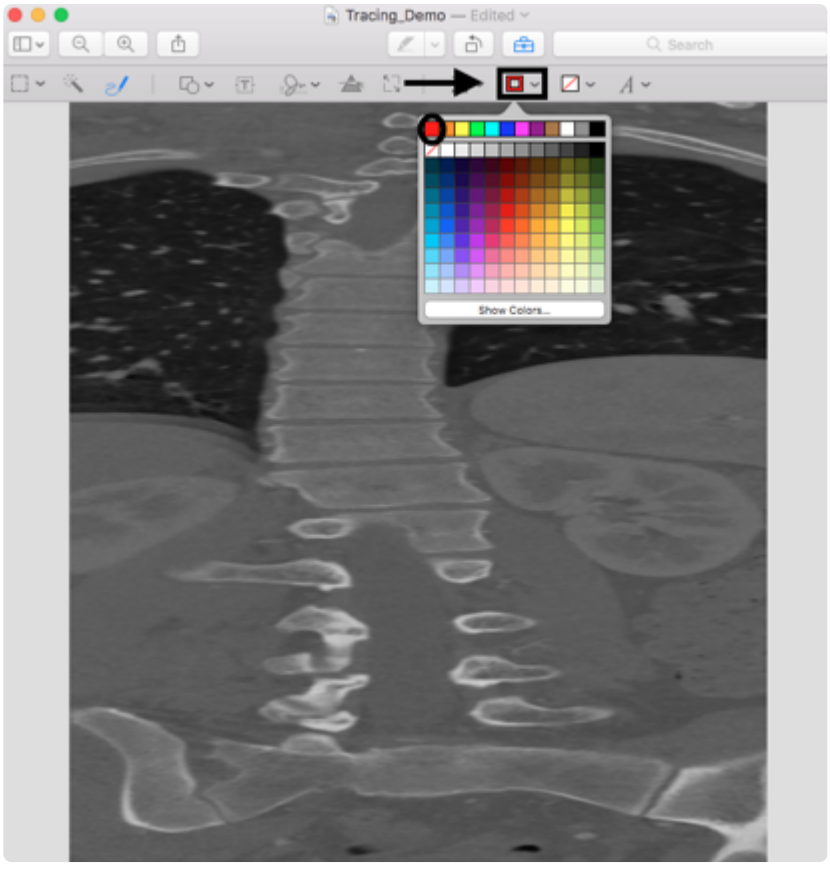
Once you've selected and opened your image, you must open the 'preview' tool box to utilize tools



Select your pen from the tool box menu

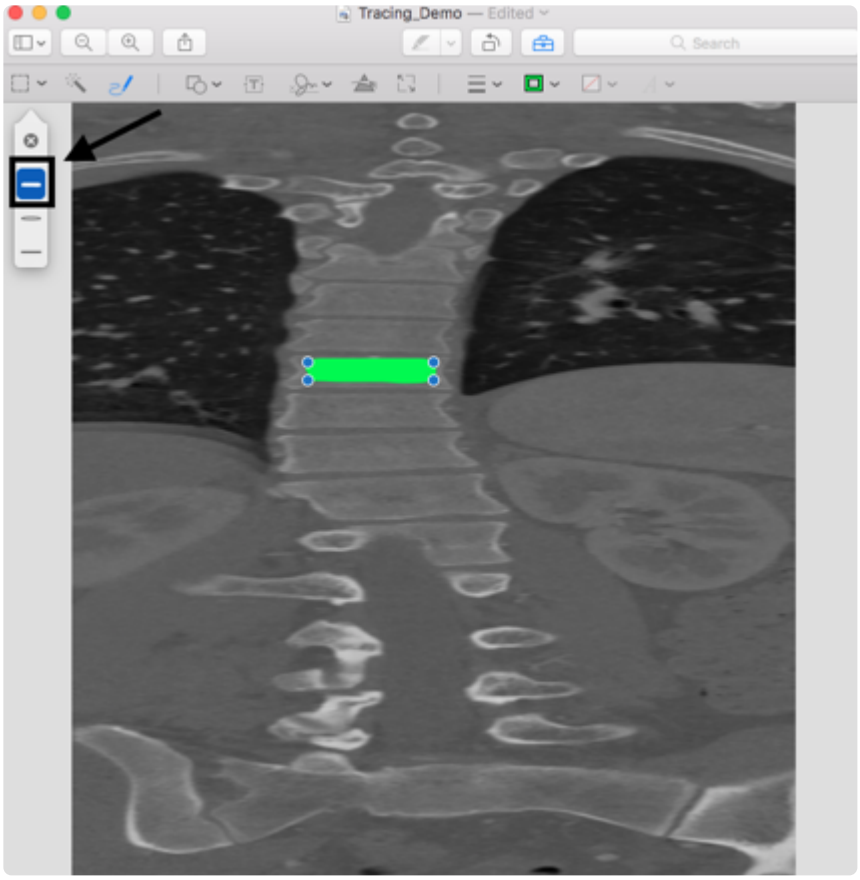


Select color ‘Red’ (Class 1), ‘Blue’ (Class 2), ‘Yellow’ (Class 3), ‘Green’ (Class 4) from the color drop down menu



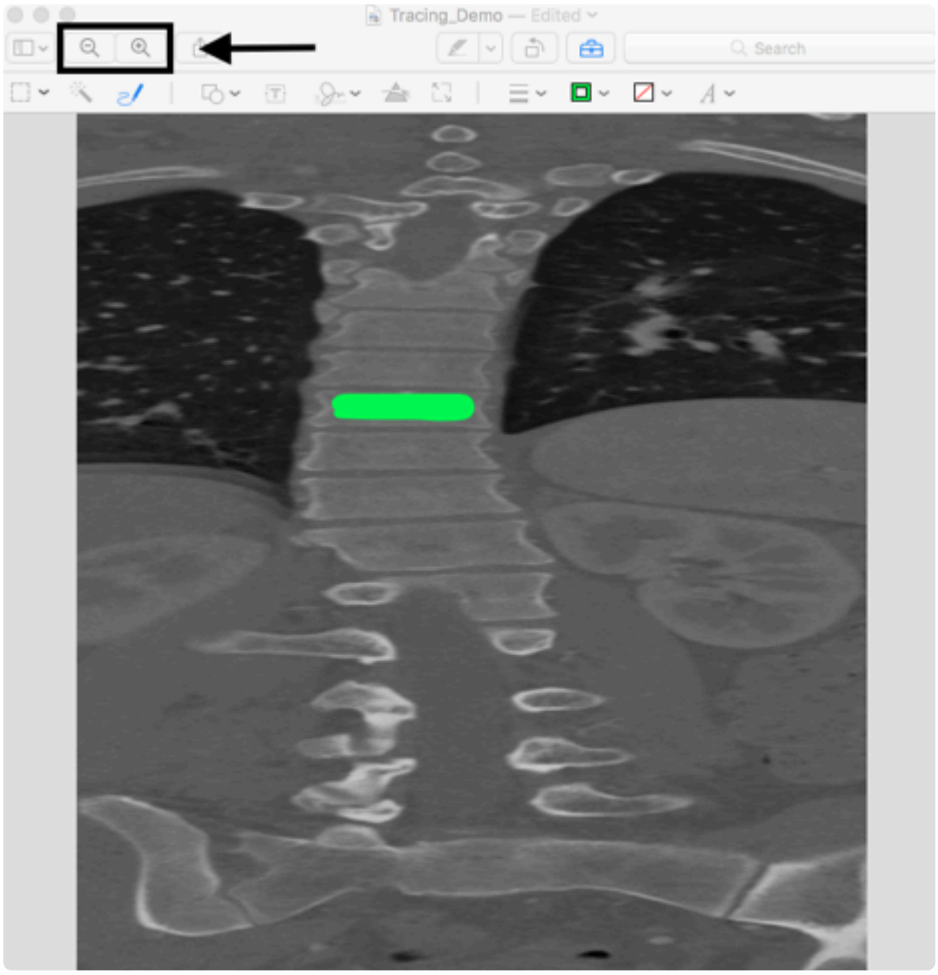
Left click and hold the house to draw/trace areas of interest

Once you release the mouse after tracing, you can select to keep manual edit instead of the auto-lines generated by ‘Preview’



Use the magnify glass icon to zoom into areas of interest. If you make a mistake just simply press the delete key or ‘Command + z’ to remove

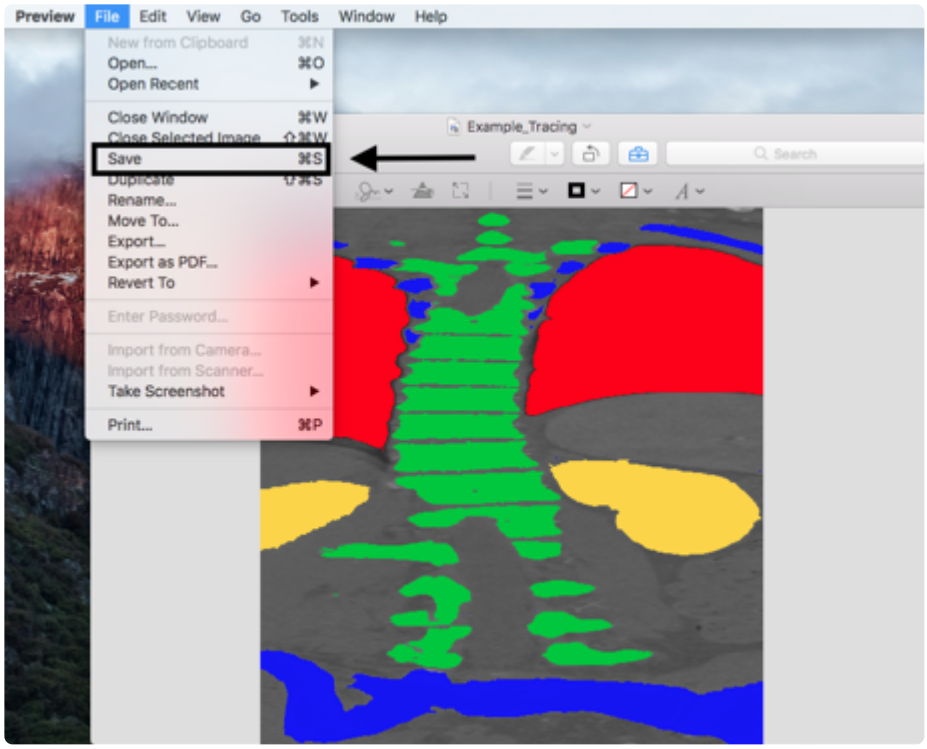
Tip: Each class should be a different item of interest, the more classes produce significantly better results i.e. background, negative classes (things we don’t want).



Save Results

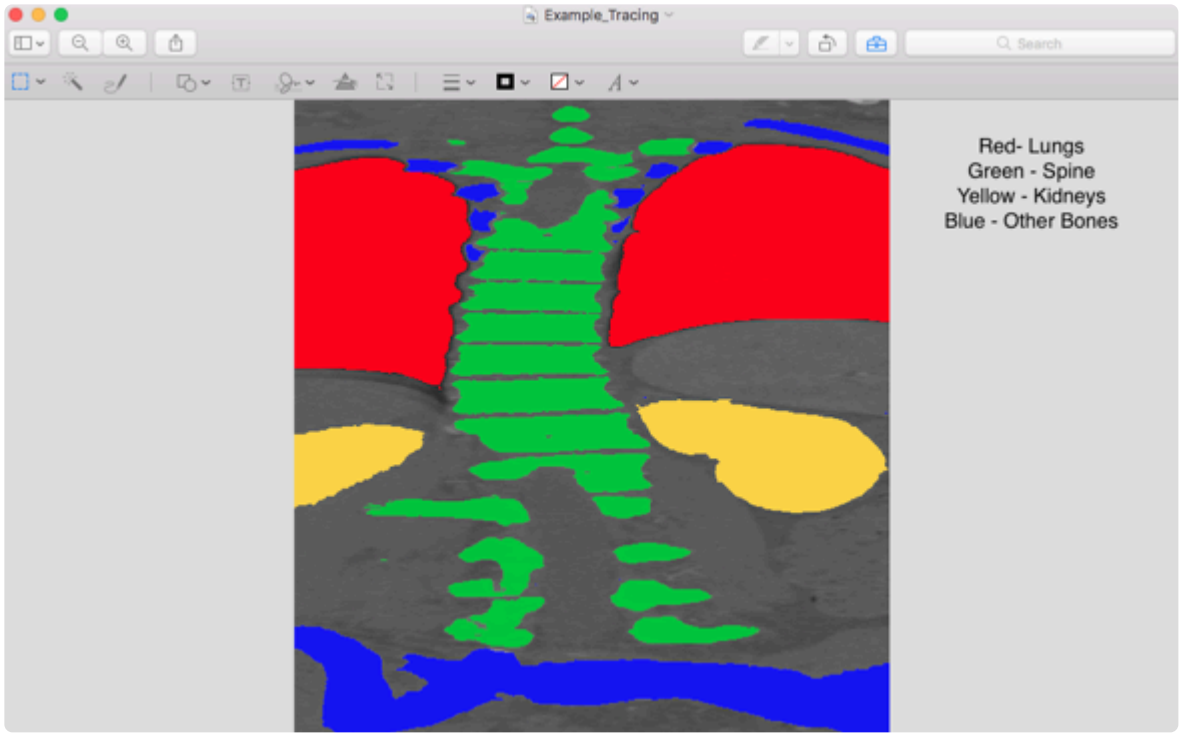
Once you're finished you can save by simply pressing file save or 'Command + s'. You may want first select 'File > Duplicate' and then use 'Command + s' save a copy of the traced image separate from the original.

Tip: Save in either TIF or PNG format to avoid blurring at the edges of your tracings.



Example Tracings

We want to identify the spine, but also selected three other classes in order to give the algorithm more data to learn from. Using these negative classes gives the software more information to know exactly what makes your desired features unique.



Import Tracings into MIPAR

Download “extract-tracings” Recipe. This can be used as a “starter-kit” for extracting color tracings from any image. The Recipe has steps to extract red and green tracings to start, but feel free to add others as needed.

Launch the **Batch Processor**.

Load the “extract-tracings.rcp” Recipe and drag and drop the set of tracings to process.

Click **Process**.

When the batch is finished, launch the **AI Session Processor**.

Click **Load Session** and choose the Session file output from the batch.

Replace the loaded Reference Images with the originals by shift-selecting all images, choosing **Replace**, and selecting all original images.

Training

Once features are traced, a deep learning model can be trained. This example shows how to train a deep learning model and assumes tracing has been completed as shown in either of the tracing examples.

Downloads

You may download the sample session file and reference images to use as learning tools and follow this example.

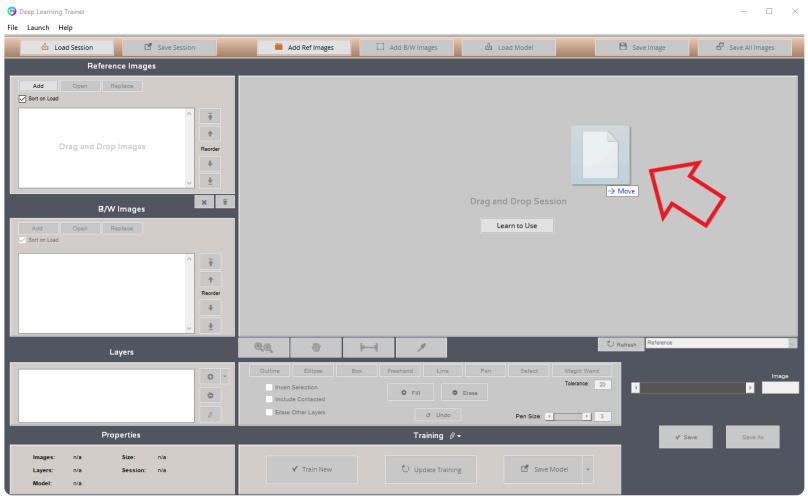
Session File (unzip > open .ssn2 file in AI Session Processor)

Reference Images

Load Images

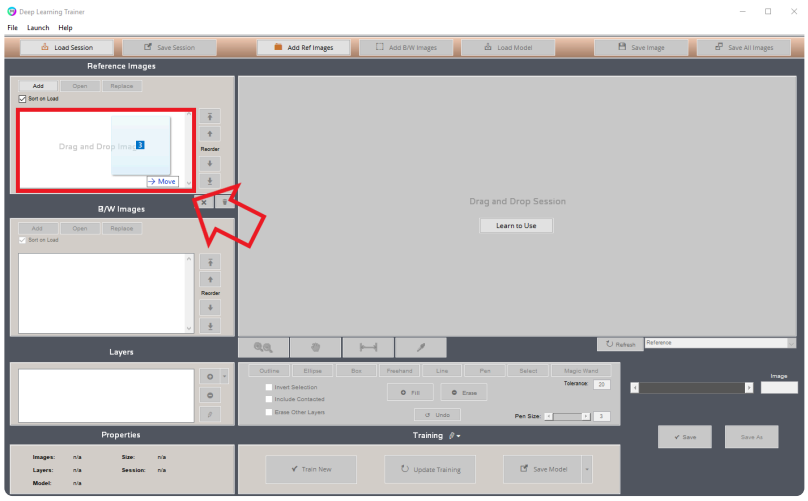
Launch the **AI Session Processor**.

If you already have a Session file saved, load it using **Load Session**, or by dragging and dropping. Session files are cross-compatible between the AI Session Processor and Session Processor.

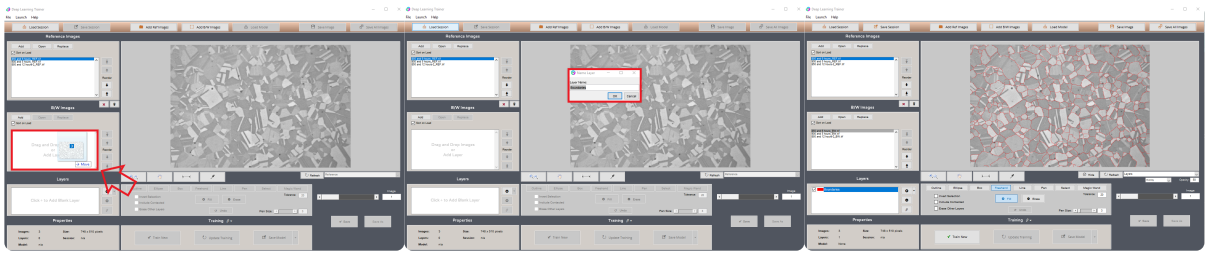


OR

Drag and drop reference images to “Reference Images” panel to add.

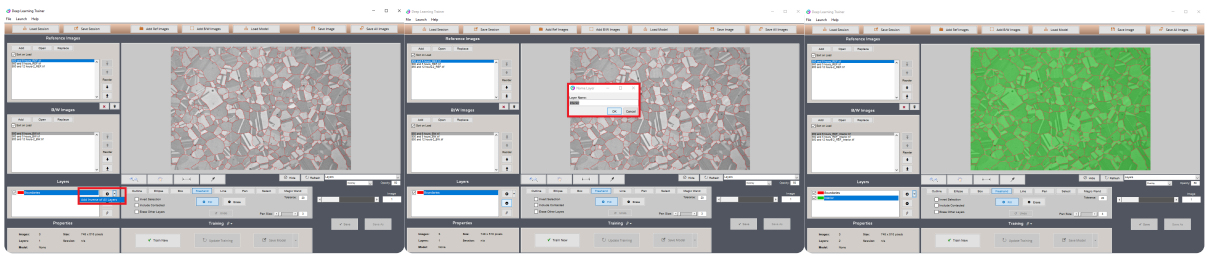


Drag and drop B/W tracings to “B/W Images” panel to add as Layer. Name Layer when prompted and click “OK”.



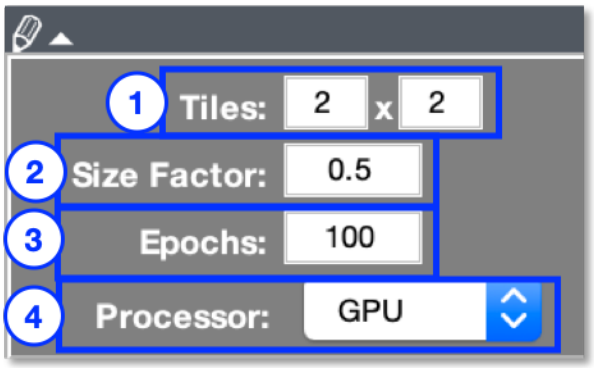
Repeat for as many Layers as needed.

If necessary, use the **dropdown arrow** next to the **+** followed by **Add Inverse of All Layers** to add a background Layer. Name the Layer and click “OK”. You’ll usually choose “No” when asked if you want to erode after inversion. This will add all undefined pixels as another Layer.



Tip: Training requires at least 2 Layers. Only use the above inversion method if all features in Layer 1 are traced. If they are not, click **+** to add the second Layer, and trace a few areas of the background.

Configure Settings



> Training images are randomly shuffled once prior to training to prevent their order biasing model convergence. Augmentation is also applied each iteration using randomized translation, rotation, and reflection.

1. Tiles

Grid spacing to split each image into tiles prior to training, defined as columns x rows. This reduces memory load on the GPU and provides more data to the training algorithm, which tends to produce better results. *(Example: Tiles = 3×2 splits each image into 6 sub-images, 3 columns across by 2 rows down)*

2. Size Factor

Resize factor for training images (0-1). This reduces GPU memory load and significantly speeds up training times. A factor up to 0.5 can often be used and still provide adequate precision.

3. Epochs

Number of epochs to run training for. The below table offers some recommendations based on your amount of training data:

Number of Training Sub-images	Epochs
< 200	500-700
200-500	300-500
500-1000	100-300
> 1000	50-100

4. Processor

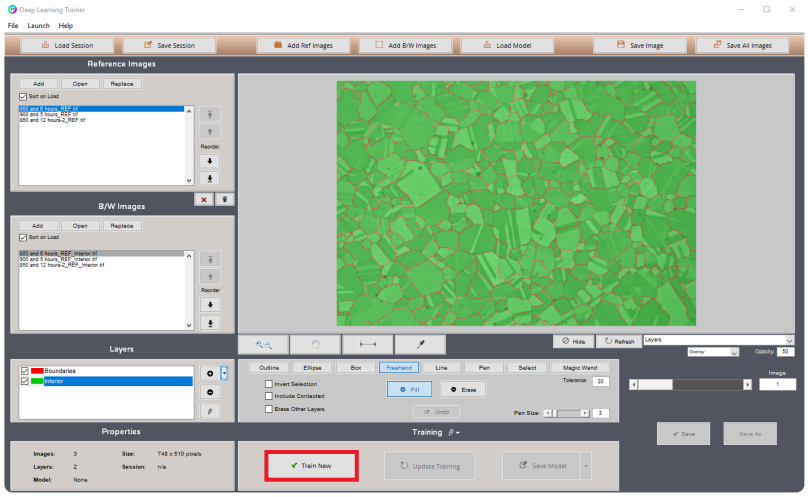
- **GPU:** Train model on single GPU (strongly recommended — see Deep Learning System Requirements for more information). Active GPU is set in “GPU Computation” panel in *File* >

Preferences.

- **CPU:** Train model on the CPU
- **Multi-GPU:** Train model on multiple GPUs on your local system (only available when more than 1 GPU is detected)

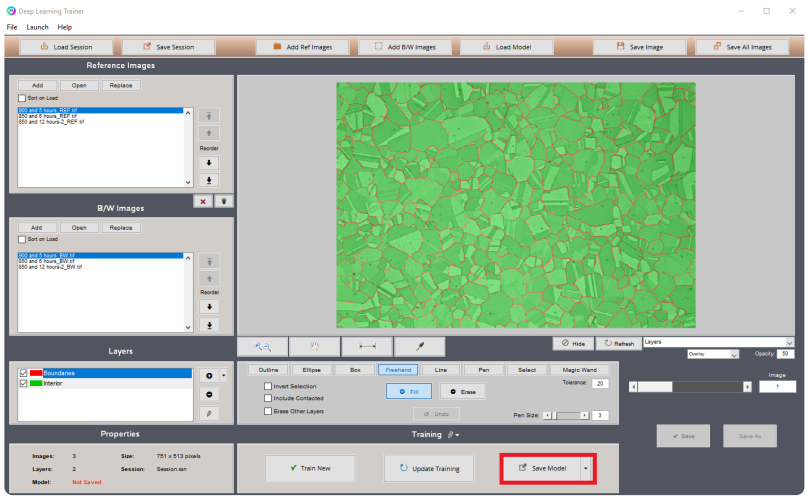
Train Model

Click **Train New** to begin training. It may take several seconds to initialize. When training begins, progress percentage and time remaining will display above the “Training” panel.



You may click **Stop Training** at any time to stop the training. The current state of the model will be preserved and you will be able to save it.

When training is complete, click **Save Model** to save model as a .DLM (MIPAR Deep Learning Model) file.



See Applying for information on applying the model.

Updating

Updating a deep learning model allows new classes to be added without retraining on the entire set that had been used up to that point.

However, our testing has shown that an extensively trained and validated model is required for updating to work properly.

Whether adding new classes or not, unless the model has been trained on at least thousands of previous images, we recommend to refine models by adding the new tracings to the original Session, and training a new model on all images using **Train New** in the AI Session Processor.

> **Tip:** When updating, make sure that your new tracings contain some features that look like those from the old images, otherwise the model may start forgetting them in favor of the new ones.

Applying

Once a deep learning model is trained, it can be applied to a new image to automate feature detection. Effectively, application of a deep learning model results in a pre-processed version of the original image, where features of interest are “lit-up” against a dark background. The rest of MIPAR’s processing library can then take over to accurately select features.

This example shows how to apply a deep learning model to a new image in the Image Processor. It assumes a model has been trained as shown in the training example. If you have not followed that example, you may download the session file below and open it in the AI Session Processor.

Downloads

You may download the sample session file, reference images, recipe, and model to use as learning tools and follow this example.

Session File (unzip > open .ssn2 file in AI Session Processor)

Reference Images

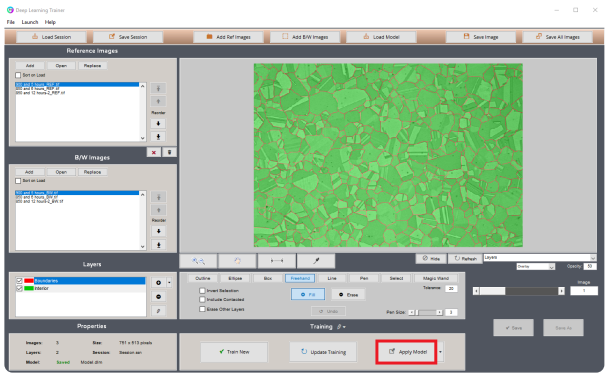
Recipe

Model

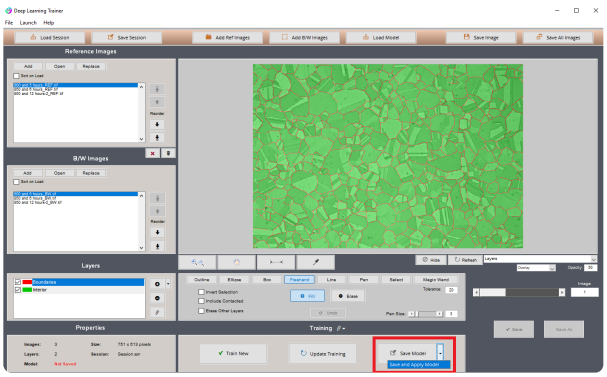
Apply Model

After training is complete and you have saved the model, you may auto-apply the model to the active Reference Image by clicking **Apply Model**.

If you have not saved the model, you may click the arrow next to “Save Model” and choose **Save and Apply Model**. This will auto-launch the Image Processor, auto-launch *Deep Learning* > ***Apply Model***, and auto-load and apply the model.



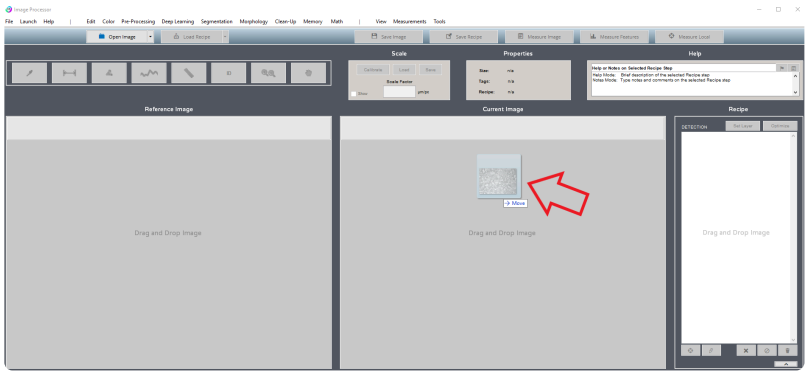
OR



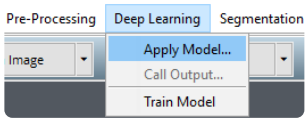
OR

Launch the **Image Processor** from the Launch Bar.

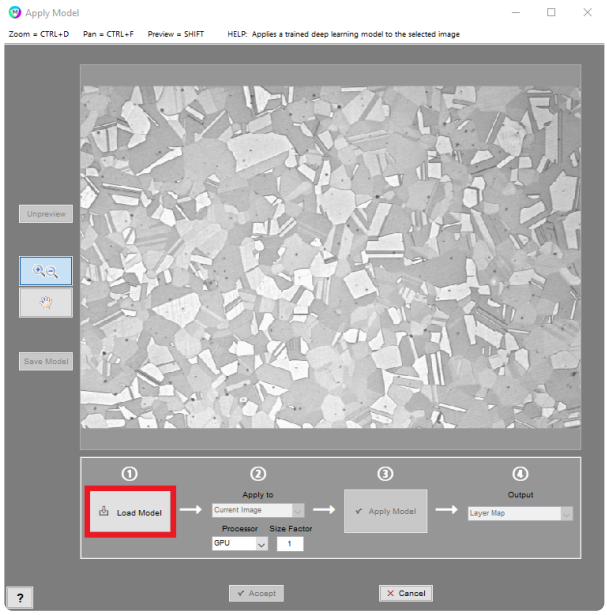
Open an image by dragging and dropping.



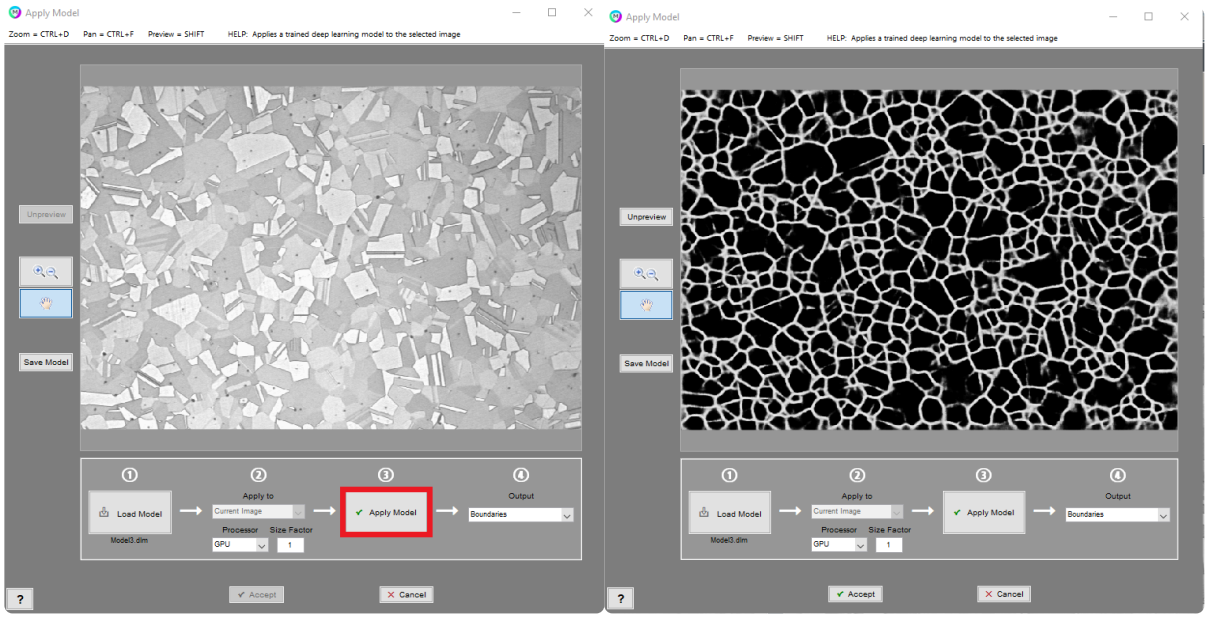
Choose *Deep Learning* > ***Apply Model***



Click **Load Model** and choose the saved .DLM file.



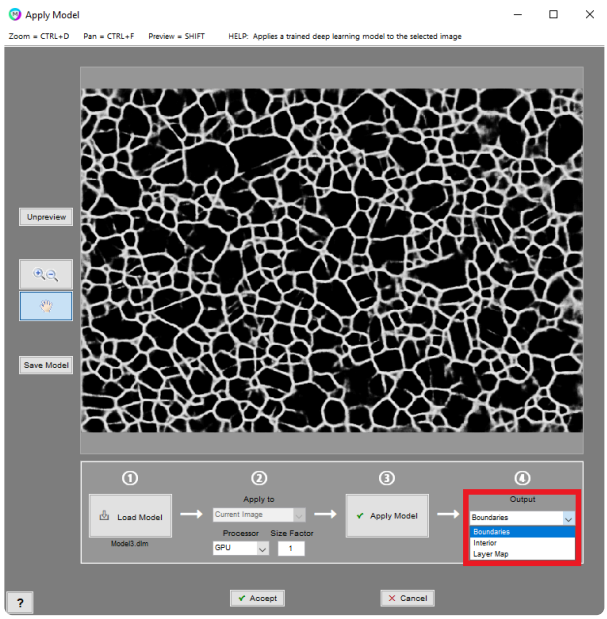
Click **Apply Model**.



> **Tip:** The “Size Factor” box is auto-populated with the size factor used to train the model. It can be adjusted if desired. See *Apply Model* documentation for more details.

Output Result

From the **Output** dropdown choose either the layer map, or one of the probability maps.

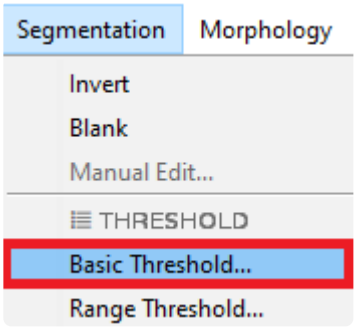


Click **Accept**.

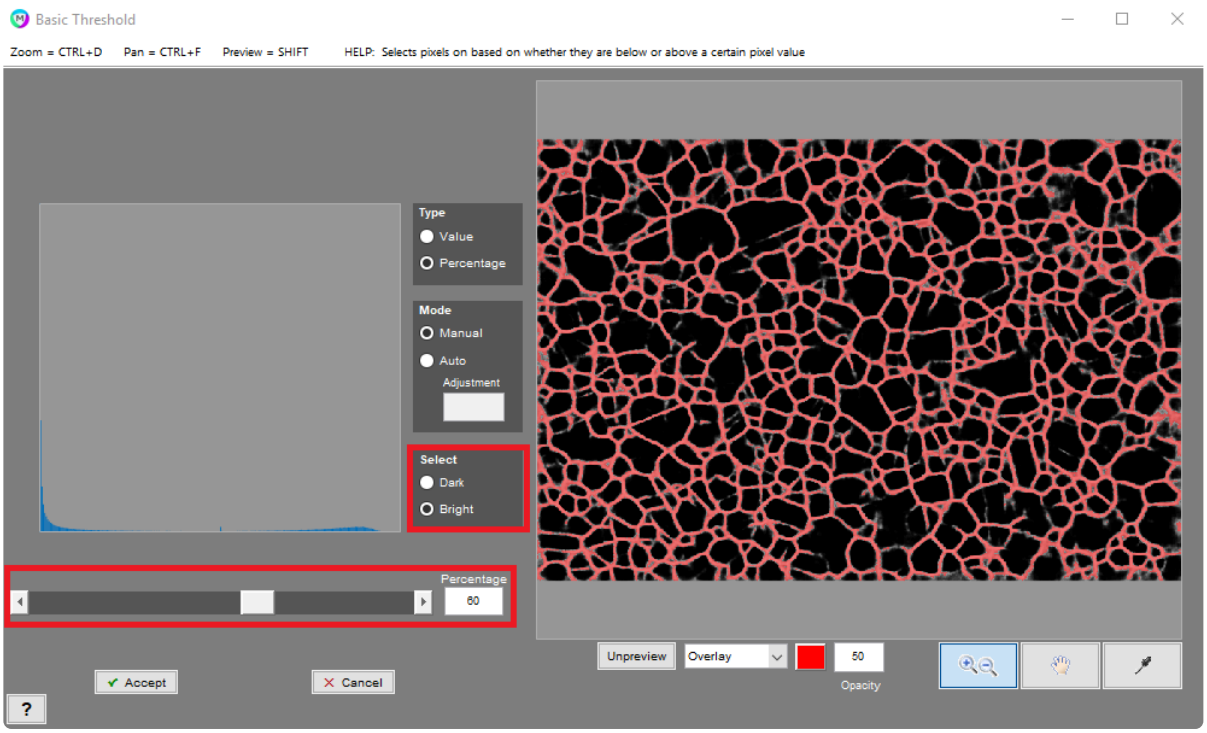
Finish Detection

> **Tip:** A key capability of working with deep learning in MIPAR is the ability to further process the probability map(s) or layer map after output. This is often critical to achieve accurate feature detection.

After outputting the probability map, select the features of interest with *Segmentation* > ***Basic Threshold***. When added after an *Apply Deep Learning* step, *Basic Threshold* will auto-launch with Type = “Percentage” and Selection = “Bright”.



Set the percentage threshold. This sets the lowest probability percentage that counts as a feature.



Click **Accept**.

Add any additional steps needed to complete the feature detection.

If you had output a probability map, use *Deep Learning* > ***Call Output*** to call other probability maps if needed.

If you had output the layer map and need to use it more than once, add a *Set Memory Image #X* after *Apply Deep Learning* and then use *Call Memory Image #X* as often as you need to select all Layers.

3D Toolbox

Requires 3D Extension

The 3D Toolbox is used for visualizing and editing Image Stacks as well as visualizing, editing, and quantifying 3D Reconstructions. Reconstructions can be exported to other visualization applications such as commercial Avizo and open source ParaView for high-end rendering and animation.

Topics

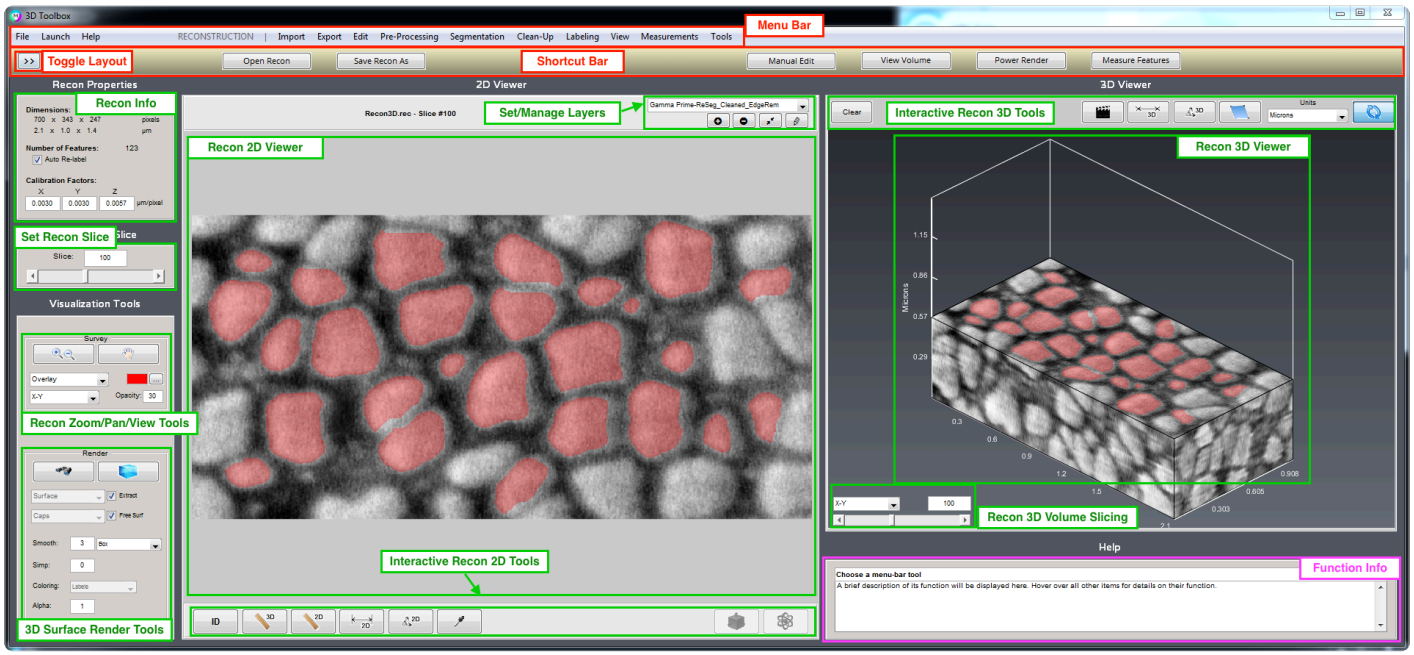
- Layout
- Reconstruction
- Visualization
- Measurements

Layout

Below are labeled screenshots of the 3D Toolbox which reveals the layouts of and purpose behind each user interface element.

Layout #1: Recon *(recommended when working with a reconstruction)*

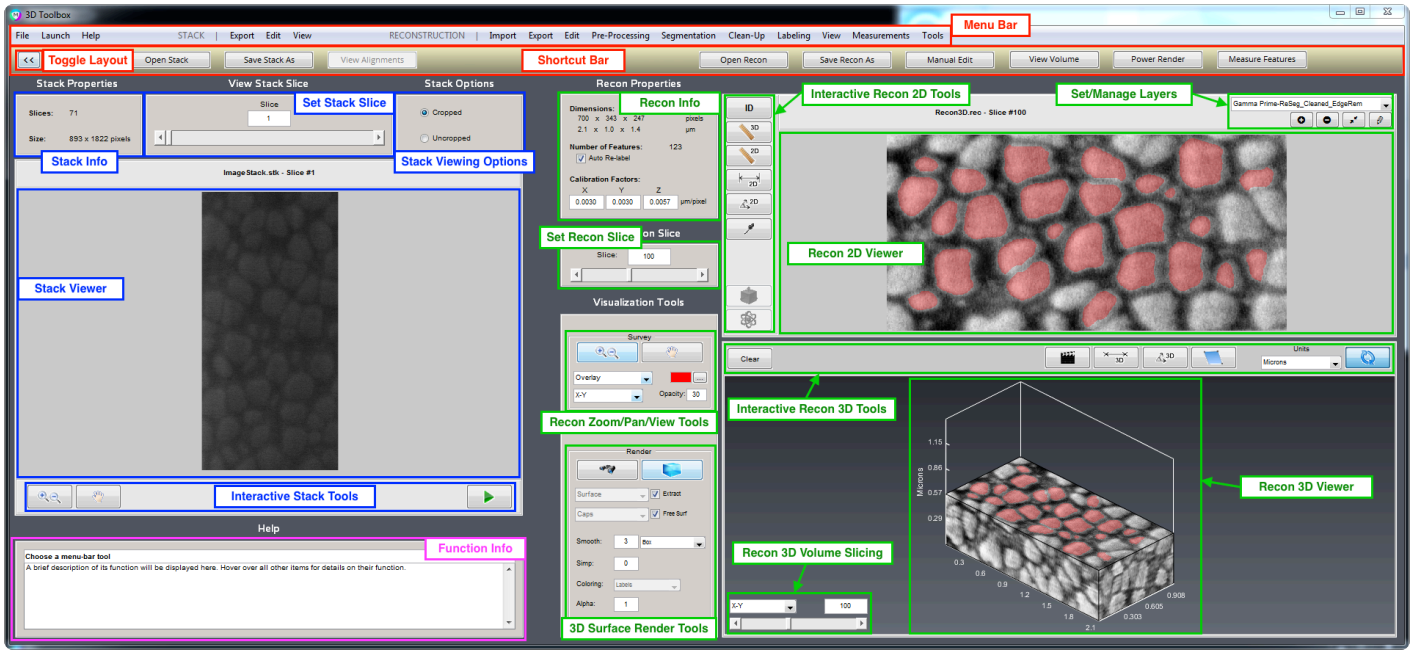
Shows the Reconstruction side only. Used when working with a Reconstruction.



> See Keyboard Shortcuts for shortcuts relevant to the 3D Toolbox.

Layout #2: Stack+Recon

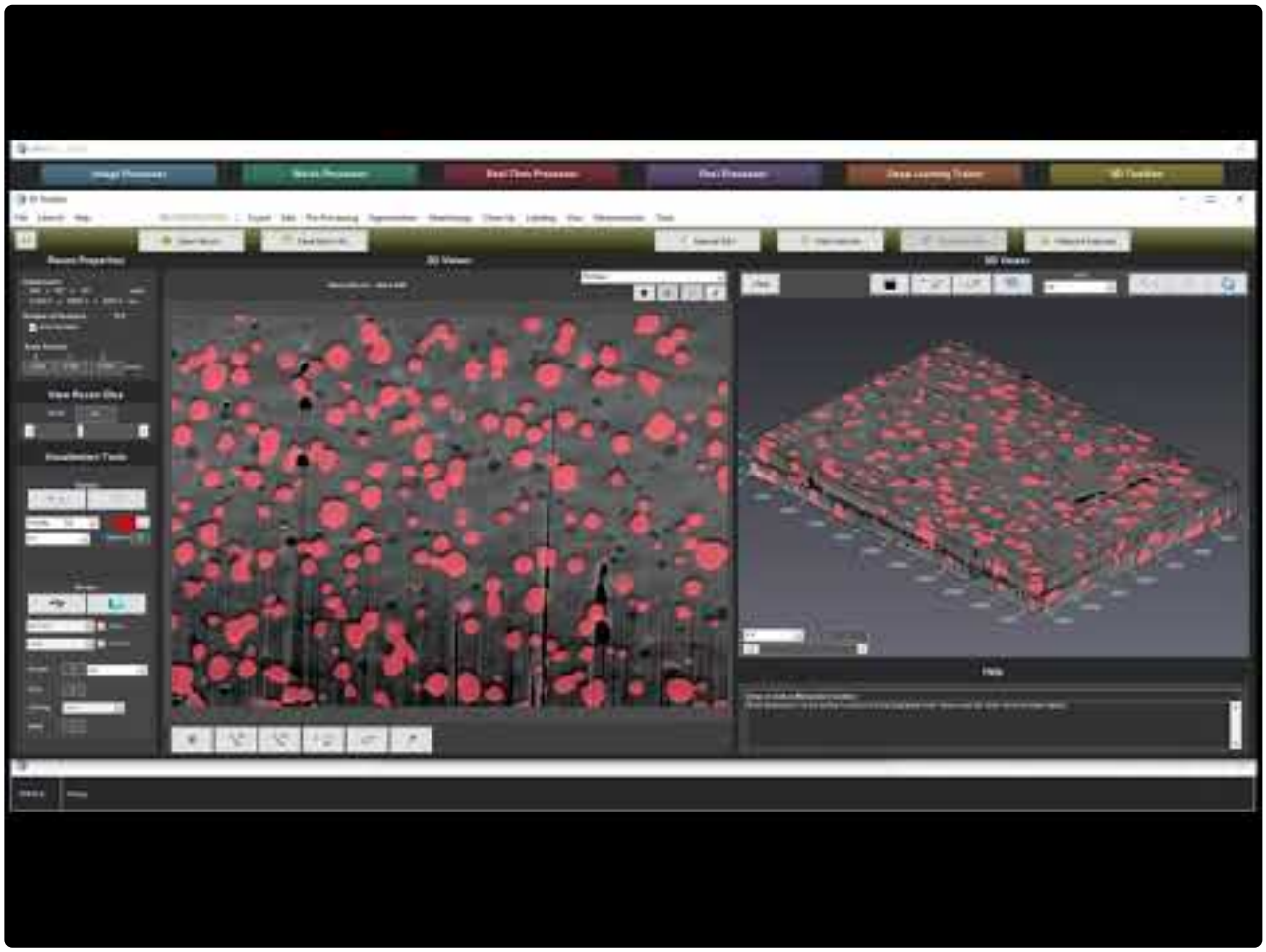
Shows both the Image Stack and Reconstruction side. Used when working with an image stack prior to segmentation and reconstruction, or when comparing the reconstruction to the aligned image stack which preceded it.


Menu Bar
Toggle Layout
Shortcut Bar
Stack Properties
View Stack Slice
Stack Options
Set Stack Slice
Stack Info
Stack Viewing Options
Stack Viewer
Interactive Stack Tools
Help
Function Info
Choose a menu-bar tool
A brief description of its function will be displayed here. Hover over all other items for details on their function.
Recon Properties
Recon Info
Set Recon Slice
Visualization Tools
Recon Zoom/Pan/View Tools
3D Surface Render Tools
Interactive Recon 2D Tools
Set/Manage Layers
Recon 2D Viewer
Interactive Recon 3D Tools
Recon 3D Volume Slicing
Recon 3D Viewer

Reconstruction

Tutorial

Shows the workflow for creating, processing, visualizing, and measure a 3D Reconstruction.



https://www.youtube.com/embed/sTMwAB9xPgQ?rel=0

https://www.youtube.com/embed/sTMwAB9xPgQ?rel=0

Downloads

You may download the below assets to use as learning tools and follow this tutorial.

Raw Dataset

Calibration Data

Segmentation Recipe

Visualization

MIPAR

MIPAR provides basic tools for visualizing your 3D Reconstruction with tools like volume slicing, surface rendering, and simple animation.

Avizo

Avizo is the leading high-performance 3D visualization and analysis software for scientific and industrial data. We recommend using Avizo for your final visualization tasks if you have access. MIPAR supports export of its surfaces, reference volumes, and segmented volumes to an Avizo file format.

More information about Avizo can be found at: https://www.fei.com/software/amira-avizo/

ParaView

ParaView is a powerful open-source visualization environment. MIPAR supports export of its surfaces, reference volumes, and segmented volumes to a VTK file format, which can be opened in ParaView.

Also, if ParaView is installed in C:\Program Files\Paraview *(not C:\Program Files\Paraview-x.x.x)* on a PC, or in “/Applications/paraview” on a Mac, the “Power Render” button in the Shortcut Bar will become active, and clicking it will export the current view of the reconstructed volume (i.e, Reference, B/W, Outline, Overlay, Labels, Layers) in a VTK file and automatically open it in ParaView.

ParaView can be downloaded at: http://www.paraview.org/download/

Measurements

Global Measurements

Global Measurements report one measurement per reconstruction. Common measurements include volume fraction of features, total surface area of features, and number density. Click any link below for descriptions of the many global measurements available in MIPAR.

- Volume
- Volume Fraction
- Surface Area
- Surface Area Fraction
- Count
- Number Density
- Intensity Mean
- Intensity StdDev
- Intensity Sum
- Intercepts

Feature Measurements

Feature Measurements report one measurement for each feature in a reconstruction. Descriptions of the many feature measurements available in MIPAR can be found here

Centroid Paths

Centroid Paths measurement generates a data table for either a selected feature or all features. The data generated consists of two tables per features (Centroid_X and Centroid Y) where the tables consist of the Z or slice position vs X or Y centroid position. File nomenclature is
‘Feature Number’_Centroid’Coordinate’

Surface Measurements

Surface Measurements report one measurement for each feature rendered. Surface render must be enabled to allow Surface Measurements. Description of the types of surface measurements can be found here

Volume

3D Measurements > Volume

Measures the total volume of selected pixels in the reconstruction.

Volume Fraction

3D Measurements > Volume Fraction

Measures the volume fraction of the selected pixels in the reconstruction. Counts all selected pixels in the active layers and divides it by the number of pixels in the layer specified in the “Relative To” window.

Surface Area

3D Measurements > Surface Area

Measures the total surface area of the selected features.

Surface Area Fraction

h.3 3D Measurements > Surface Area Fraction

Surface Area Fraction measures the surface area of all the selected features in the active layer and divides it by the surface area of the layer specified in the ‘Relative To’ window.

Count

3D Measurements > Count

Reports how many selected features are present in the reconstruction.

Number Density

3D Measurements > Number Density

Number Density measures the number of features per unit volume, where the total volume is specified in the ‘Relative To’ window.

Intensity Mean

3D Measurements > Intensity Mean

Measures the average grayscale intensity within selection.

Intensity StdDev

h.3 3D Measurements > Intensity StdDev

Measures the standard deviation of grayscale intensity within selection.

Intensity Sum

3D Measurements > Intensity Sum

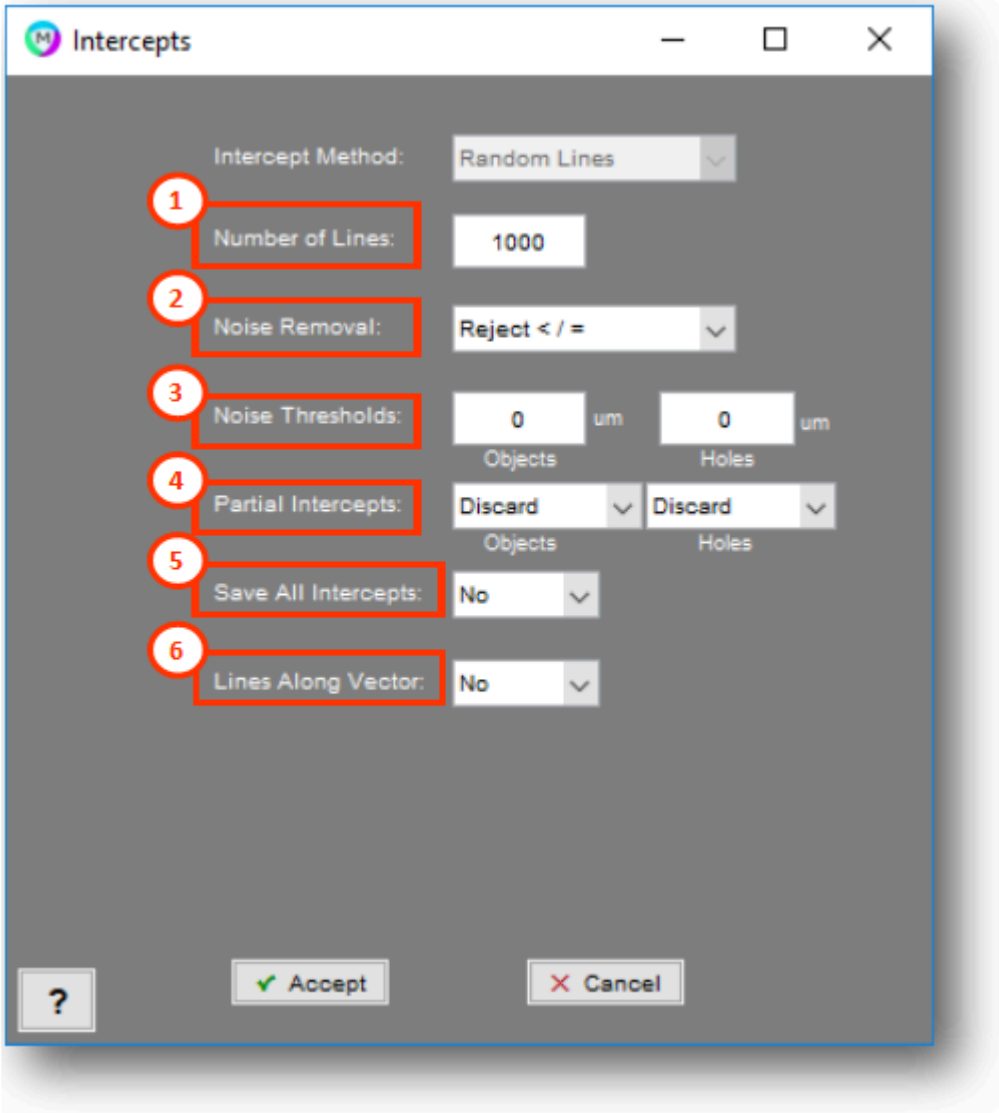
Measures the total of grayscale intensity within selection.

Intercepts

3D Measurements > Intercepts

Random Lines

Measures metrics such as mean intercept by drawing a specified number of random lines through the features in the reconstruction.



1. Number of Lines

Number of lines to draw

2. Noise Removal

Method for removing small intercepts

- **Reject < / =:** Simply ignores intercepts below a certain length
- **Fit to Noise:** *(Only for "Random Lines")* Fits an exponential to the first few bins and subtracts this exponential out the intercept histogram.

3. Noise Thresholds

Thresholds for either the minimum intercept length (for "Reject </="), the number of bins to use for "Fit to Noise"

4. Partial Intercepts

Choose whether to discard or keep intercepts that start or end at an image edge

5. Save All Intercepts

Choose whether to save all intercept lengths into text files

6. Lines Along Vector

Choose whether to draw all "Random Lines" along a specific direction. Starting points of each line will still be random.

Measurements

- **Mean Intercept – Objects:** Average length of intercepts which pass through selected features (objects)
- **Mean Intercept – Holes:** Average length of intercepts which pass through un-selected areas (holes)
- **Mean Inverse Intercept – Objects:** Average inverse length of intercepts which pass through selected features (objects)
- **Mean Inverse Intercept – Holes:** Average inverse length of intercepts which pass through un-selected areas (holes)
- **Mode Intercept – Objects:** Mode (most common) length of intercepts which pass through selected features (objects)
- **Mode Intercept – Holes:** Mode (most common) length of intercepts which pass through un-selected areas (holes)
- **Mode Inverse Intercept – Objects:** Mode (most common) inverse length of intercepts which pass through selected features (objects)
- **Mode Inverse Intercept – Holes:** Mode (most common) inverse length of intercepts which pass through un-selected areas (holes)
- **Total Line Length:** Total length of all lines which are drawn in order to collect intercepts.

- **Total Intersections:** Number of intersections produced between the drawn lines and intercepted features.

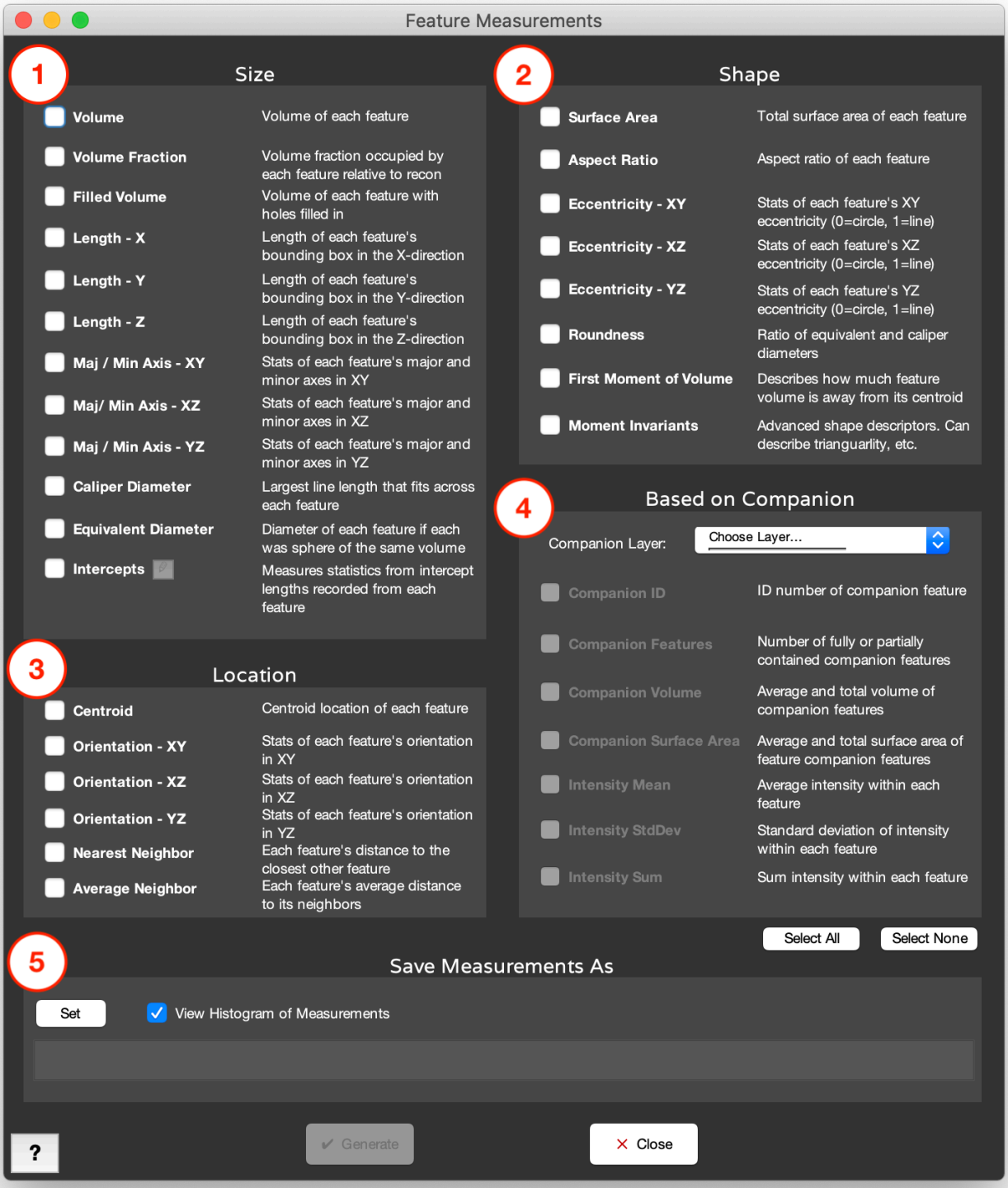
References

[1] ASTM E112-13, Standard Test Methods for Determining Average Grain Size, ASTM International, West Conshohocken, PA, 2013, www.astm.org

Feature Measurements

3D Measurements > Feature Measurements

Feature Measurements allows a variety of measurements to be made from each feature in the reconstruction or Layer.



1. Size

- **Volume**: Volume of each feature.
- **Volume Fraction**: Volume fraction occupied by each feature relative to entire reconstruction volume.
- **Filled Volume**: Volume of each feature with holes filled in.

- **Length – X**: Length of each feature's bounding box in the X-direction
- **Length – Y**: Length of each feature's bounding box in the Y-direction.
- **Length – Z**: Length of each feature's bounding box in the Z-direction.
- **Maj/Min Axis – XY**: Stats of each feature's major and minor axis in the XY plane.
- **Maj/Min Axis – XZ**: Stats of each feature's major and minor axis in the XZ plane.
- **Maj/Min Axis – YZ**: Stats of each feature's major and minor axis in the YZ plane.
- **Equivalent Diameter**: Diameter of each feature if each was a sphere of the same volume.
- **Caliper Diameter**: Largest line length that fits within each feature.
- **Intercepts**: Measures statistics from intercept length recorded from each feature.
  - **Through Centroid**: Draw each line through each feature's centroid.
  - **Along Vector**: Draw each line along the same direction through each feature.

2. Location

- **Centroid**: Centroid location of each feature.
- **Orientation – XY**: Stats of each feature's orientation in the XY plane, where the angle is calculated by replacing the XY section with an ellipse [1] and measuring its major axis orientation relative to the X-axis.
- **Orientation – XZ**: Stats of each feature's orientation in the XZ plane, where the angle is calculated by replacing the XZ section with an ellipse [1] and measuring its major axis orientation relative to the X-axis.
- **Orientation – YZ**: Stats of each feature's orientation in the YZ plane, where the angle is calculated by replacing the YZ section with an ellipse [1] and measuring its major axis orientation relative to the Y-axis.
- **Nearest Neighbor**: Each feature's distance to the closest other feature, calculated from the features' centroids.
- **Average Neighbor**: Each feature's average distance to its neighbors, as defined by the features' Delaunay triangulation. Triangulation and distance are both calculated from the features' centroids.

3. Shape

- **Surface Area**: Total surface area of each feature.
- **Aspect Ratio**: Magnitude of the largest principal axis, divided by the average of the remaining axes
- **Eccentricity – XY**: Describes how elongated or circular each feature is in the XY plane per Z section. 0 is a perfect circle. 1 is a straight line.
- **Eccentricity – XZ**: Describes how elongated or circular each feature is in the XZ plane per Y section. 0 is a perfect circle. 1 is a straight line.
- **Eccentricity – YZ**: Describes how elongated or circular each feature is in the YZ plane per X section. 0 is a perfect circle. 1 is a straight line.
  - Eccentricity is calculated from the equivalent ellipse as *sqrt(1-([minor axis length]/[major axis length])^2)* [1]

- **First Moment of Volume** *: Describes how much volume is extended away from each feature's centroid. Calculated as the summation of (volume x distance from centroid) over all voxels in the feature.

- **Moment Invariants**: High-order moments which describe different shape properties of each feature [2,3].

4. Based on Layer

- **Companion ID**: Feature ID of the Companion that contains the Current feature being measured.
- **Companion Features**: Number of features in the Companion layer that are in the Current layer.
- **Companion Volume**: Average and total volume of features in the Companion layer that is in the Current layer.
- **Companion Surface Area**: Average total surface area in the Companion layer that is in the Current layer.
- **Intensity Mean**: Average grayscale intensity within each feature.
- **Intensity StdDev**: Standard deviation of grayscale intensity within each feature.
- **Intensity Sum**: Sum grayscale intensity within each feature.

5. Save Measurement File As

The “Set” button will allow you to choose where you would like the measurement results to be saved. Check “View Histogram of Measurements” to view results as a histogram after generation.

References

[1] Ellipse is fit to the feature as the ellipse with the same second moments as the feature. Second moments are calculated as:
uxx = sum(x^2)/N + 1/12
uyy = sum(y^2)/N + 1/12
*uxy = sum(x*y)/N*

where x is the x coordinate of each pixel in the feature, y is the y coordinate of each pixel, and N is the number of pixels.

The major and minor axes are calculated as:

*common = sqrt((uxx – uyy)^2 + 4*uxy^2)*
*MajorAxisLength = 2*sqrt(2)*sqrt(uxx + uyy + common*
*MinorAxisLength = 2*sqrt(2)*sqrt(uxx + uyy – common)*

[2] Rosin, Paul L. 2003. “Measuring Shape: Ellipticity, Rectangularity, and Triangularity.” Machine Vision and Applications 14 (3): 172–184.

[3] MacSleyne, J P, J P Simmons, and M De Graef. 2008. “On the Use of Moment Invariants for the Automated Analysis of 3D Particle Shapes.” Modelling and Simulation in Materials Science and Engineering 16 (4) (June 1): 045008.

Slice-by-Slice Measurements

Measurements > Slice-by-Slice Measurements

Slice-by-Slice Measurements allows a variety of measurements to be made from each feature in each X-Y slice in the reconstruction. The 3D feature labeling is retained throughout the slices.

Details of available measurements are identical to those in 2D Feature Measurements.

Surface Measurements

3D Measurements > Surface Measurements

Surface Measurements allow you to make measurements from the currently rendered surface(s).

> Surface measurements can take some time depending on the resolution and feature density of your rendered surface(s). It is recommended that you render your surface with 50% or more simplification prior to taking surface measurements. Enter the desired simplification percentage in the “Simp” box in the Render Panel prior to generating your surface.

- **Positions**: Measures the X, Y, Z coordinates of each point on the surface
- **Curvatures**: Measures the local radius curvature (in um) at each point on the surface [1]. The averaging radius controls the extent (in um) to which the surface positions are averaged prior to the local curvature calculation.
- **Thicknesses**: Measures at each point on the surface, the distance to the other side along that point’s local normal
- **Normals**: Measures the local normal at each point on the surface

References

[1] David Cohen-Steiner and Jean-Marie Morvan. Restricted Delaunay triangulations and normal cycle. In Proc. 19th Annual ACM Symposium on Computational Geometry, pages 237-246, 2003.

Report Generator

Requires Report Extension

The Report Generator is used to export MIPAR results into a Microsoft Office Word document.

Topics

System Requirements

Microsoft Office Word

- Microsoft Office Word 2003 – Microsoft Office 365

Layout

Report Types

Available report types are as follows:

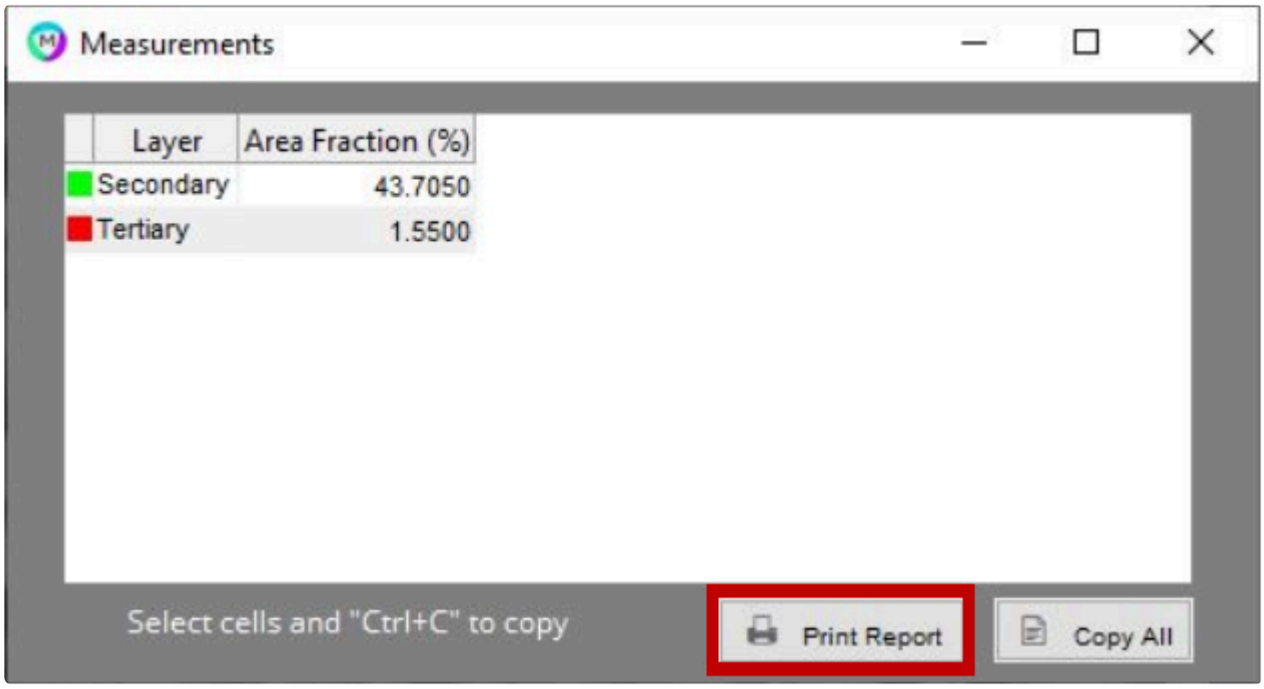
Report Type	Generated From
Single Image Global	Global measurement table in Image Processor
Single Image Feature	Color by Measurements
Single Image Local	Local Measurements
Batch Image Global	Global measurement table in Post Processor
Batch Image Feature	Histogram of Measurements opened by Post Processor

Generate Report Buttons

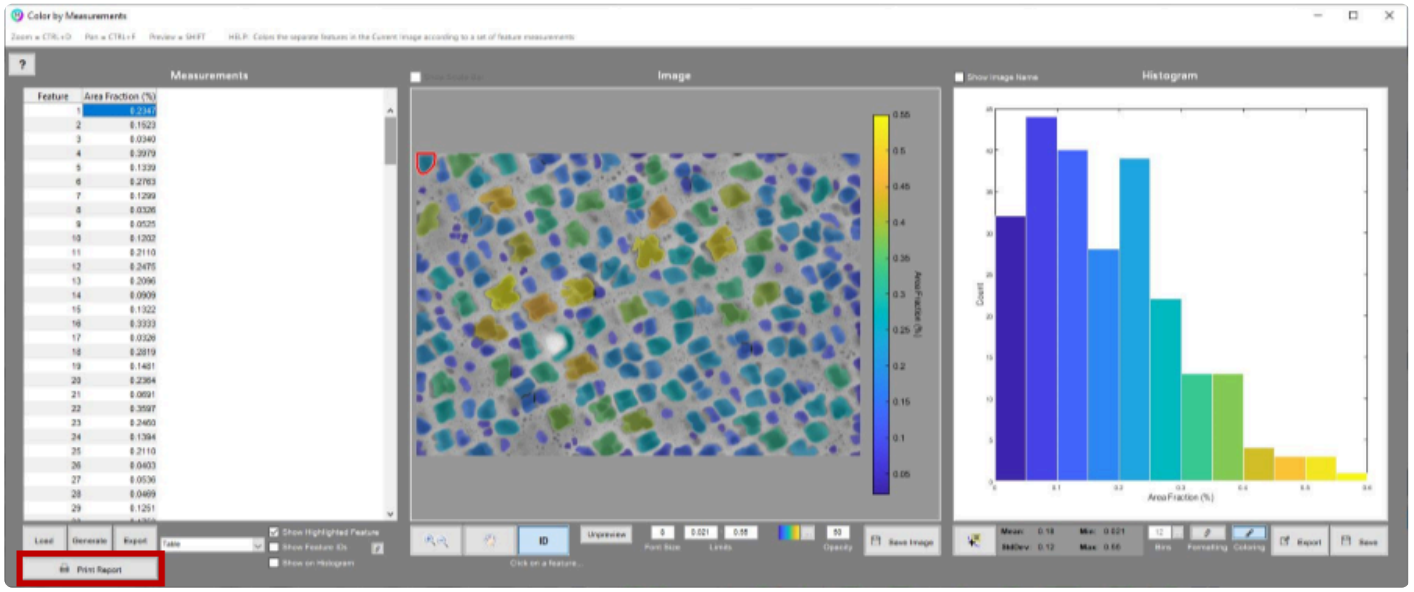
User can find the option to generate a report in most measurement windows.

Image Processor

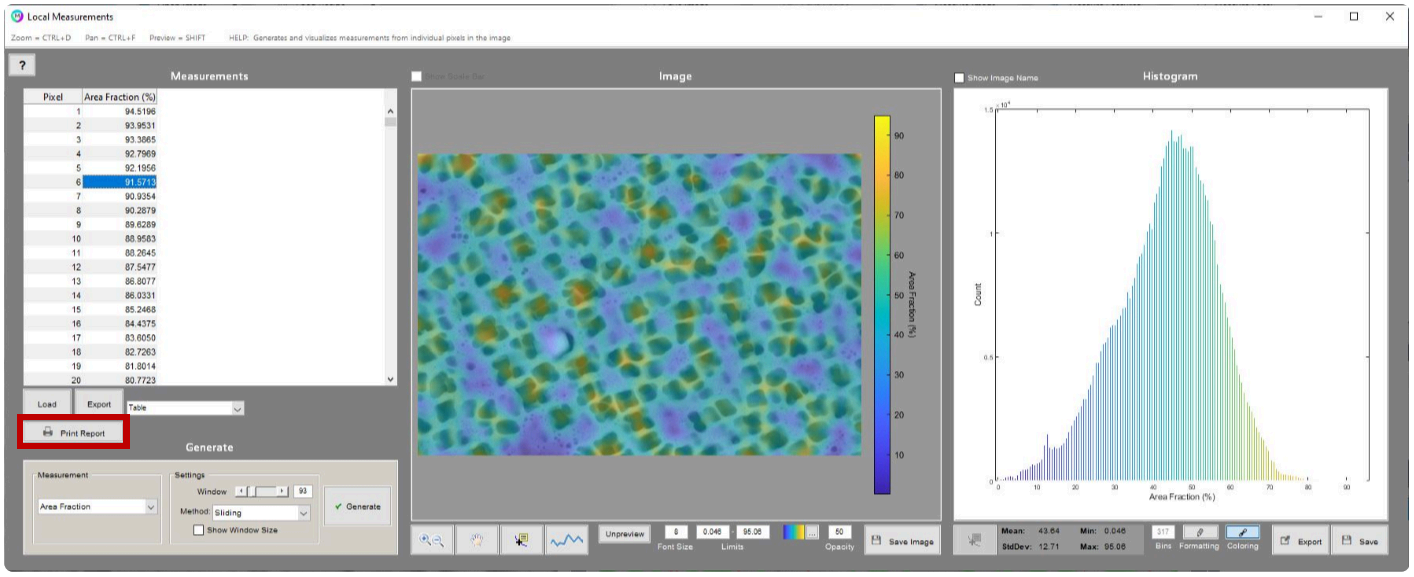
Single Image Global



Single Image Feature

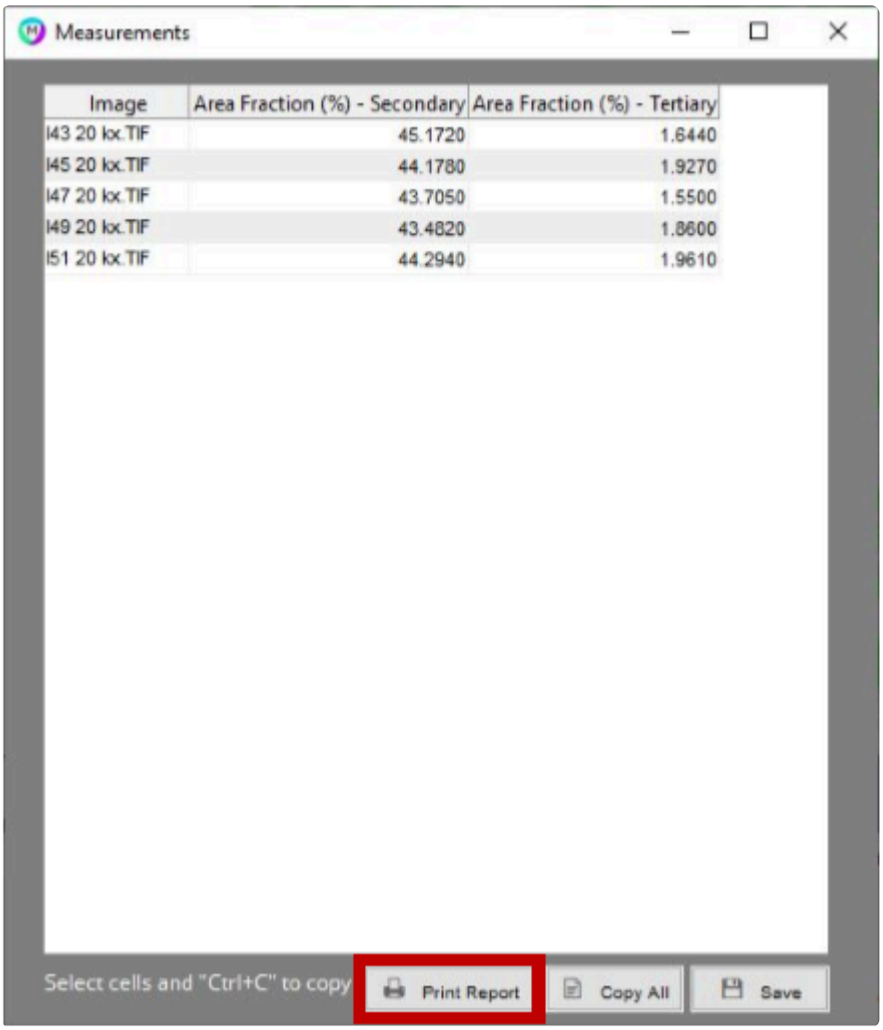


Single Image Local

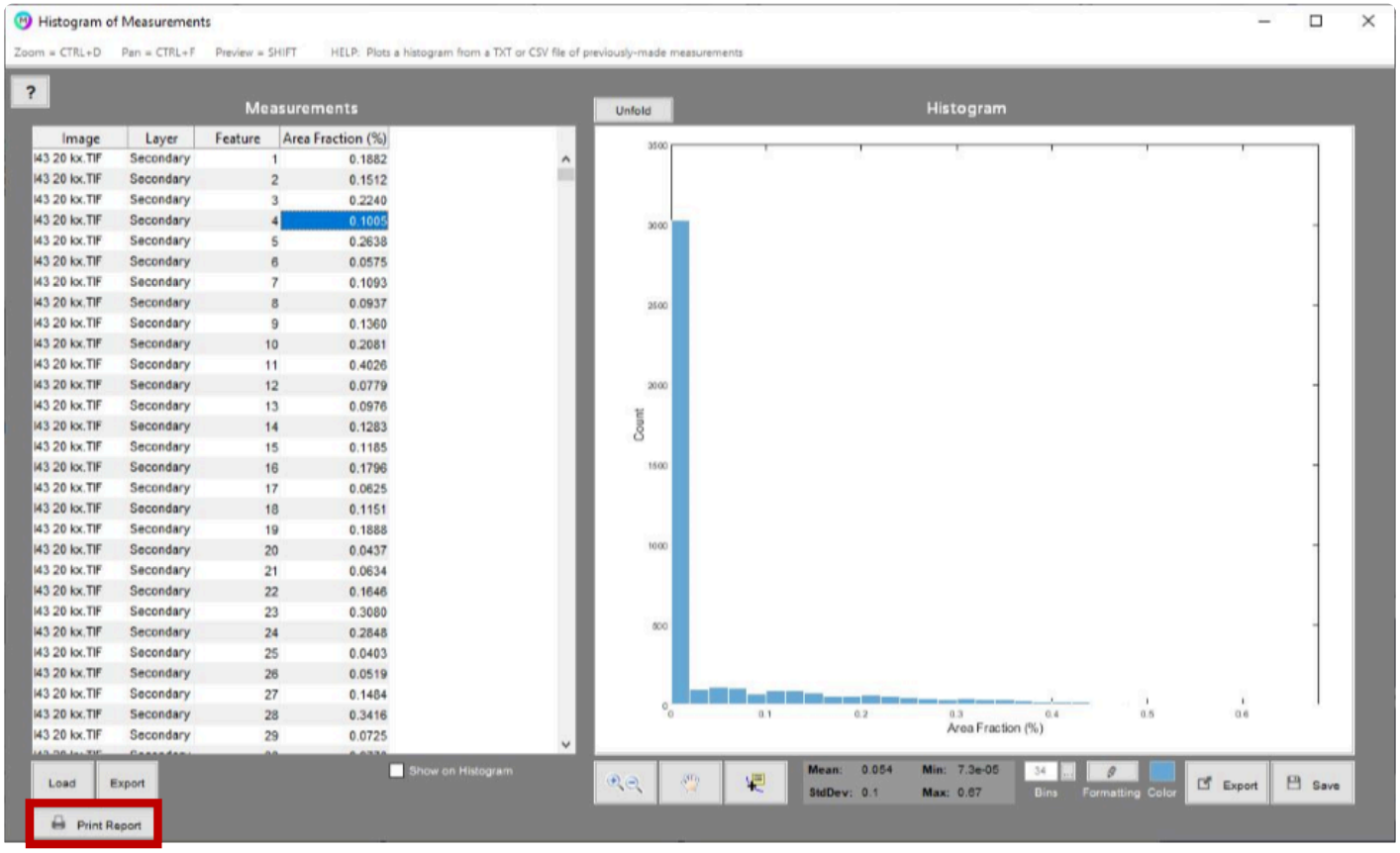


Post Processor

Batch Image Global



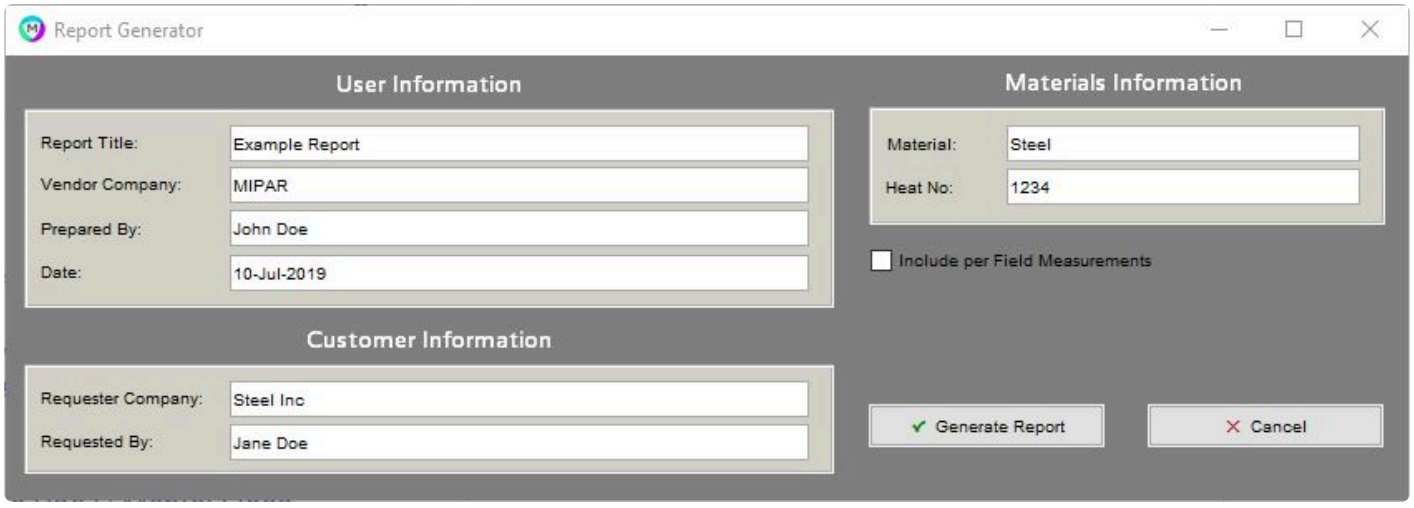
Batch Image Feature



User Information

Below is a screenshot of the Report Generator interface. The information entered into the fields will be entered into the header of the report.

Report Generator Interface



Generated Report

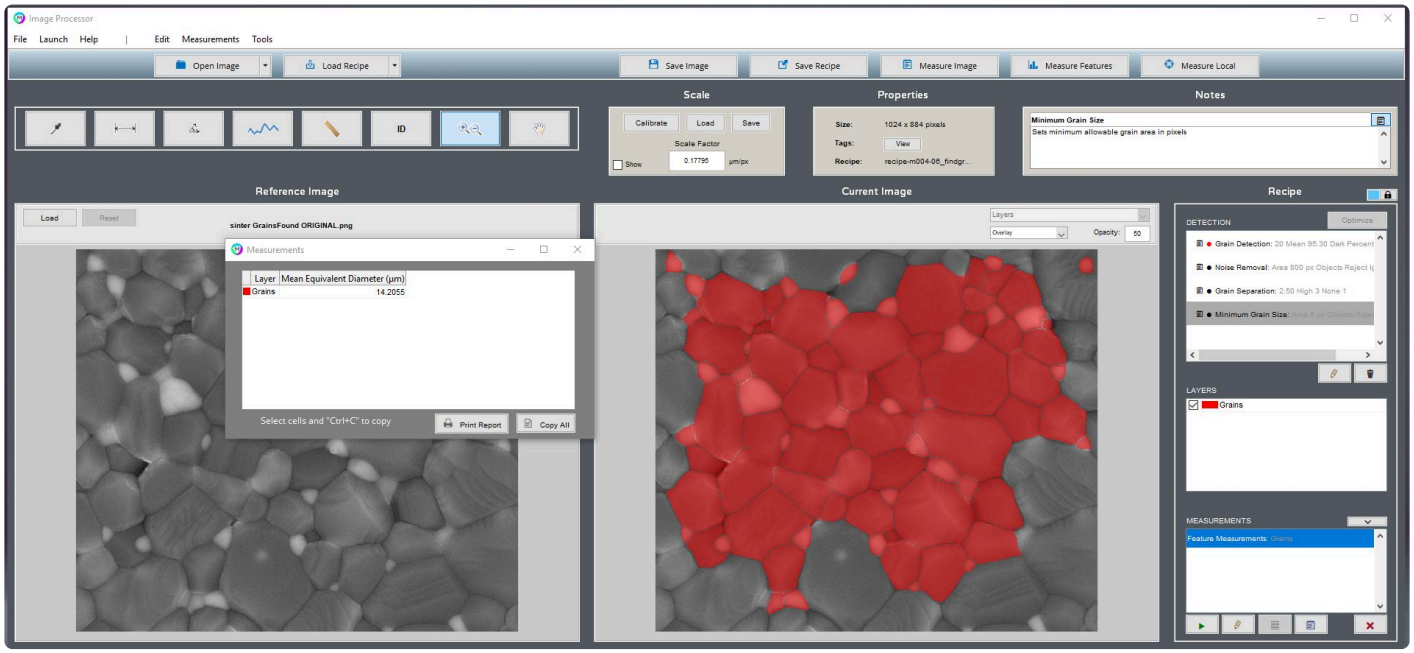
Report Title: Example Report
Vendor: MIPAR
Prepared By: John Doe
Date: 10-Jul-2019

Requester Company: Steel Inc
Requested By: Jane Doe
Material: Steel
Heat No: 1234

Report Data

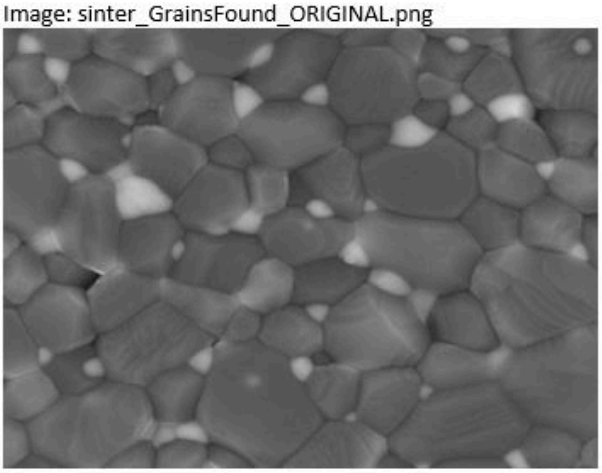
Below is a screenshot demonstrating how the data setup in MIPAR will be transferred into the Word report.

MIPAR Interface

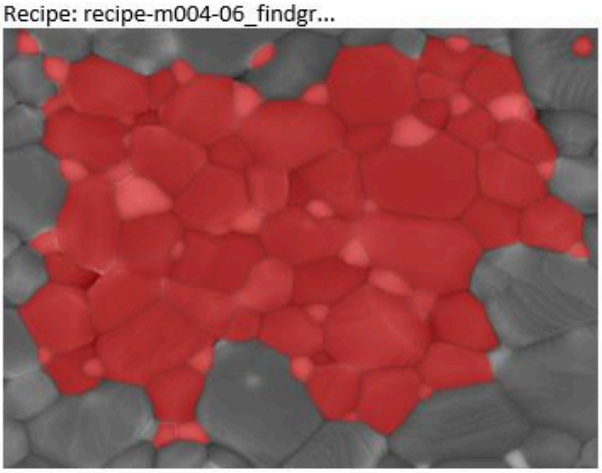


Generated Report

Image: sinter_GrainsFound_ORIGINAL.png



Recipe: recipe-m004-06_findgr...



Grains

MIPAR Version: 3.0.0
Scale Factor: 0.17795
Units: µm
Modified last: 14-May-2019 13:40:22

Measurements

Layer	Grains
Mean Equivalent Diameter (µm)	14.2055

Comments:

Signature: _____

MIPAR
Image Analysis Software

1 | Report

Templates

Report Generator Templates

The report generator templates can be found in the MIPAR install directory.
Default directory: C:\Program Files\MIPAR\reports

Editing Templates

Users can rearrange and add content to the templates. With some restrictions:

- Do not move fields in the header of the template into the body of the template
- Do not rename the templates
- When making changes to the templates, save a copy of the original template under a different file name

Adding a Company Logo

Users can add a Logo/Image to the templates in the header in the ‘companyLogo’ field, right click on the field and select ‘Remove Content Control’. Save the document as a Word Template with the same name.

Python API

Welcome to MIPAR Python API documentation. This documentation will describe the library capabilities and outline some example workflows.

Topics

Introduction

MIPAR API is a powerful tool that packages the MIPAR engine into a Python library.

Scope
The MIPAR API was developed to meet the needs of customers who are already using MIPAR to do powerful image analysis, but would like to setup “headless” workstations that execute algorithms with minimal supervision. The library is in Python as it is extremely versatile and widely adopted.

Capabilities
At this time the MIPAR API is only able to execute pre-built MIPAR recipes, this means that you must have an already developed MIPAR recipe to utilize the MIPAR API. The engine accepts images, recipes and calibration factors and returns the resulting binary masks and global measurements. See Library for more detail.

Install and Activate

Standard Install Library

1. Download the installation package provided to you by the MIPAR support team.
2. Install python 3.7,3.8 or 3.9
3. Add python to Windows path
4. Run MIPAR-API_v.x.x.x_InstallPack_Win.exe. Follow extraction and installation wizard instructions.
5. The installer will install a miparAPI folder in Program Files and attempt to install the miparAPI package into the default python environment
6. Add a new system environment variable: `miparAPI` with value: `<Path to the miparAPI folder in your Python Environment>`

Activate Trial License

1. Activate trial license.v2c file on the API system Link

Activate Local License

1. Activate local license: Link

Activate Network License

1. Install server side license manager Link
2. Configure client side license manager on machine that will be running the API Link

> It might take some time for the server side license manager to see the client activation, please be patient.

Custom Environment Install

1. Run MIPAR-API_v.x.x.x_InstallPack_Win.exe. Follow extraction and installation wizard instructions.
2. The installer will install a miparAPI folder in Program Files and attempt to install the miparAPI package into the default python environment
3. Copy the miparAPI folder in the site-packages folder to the packages folder of the new environment
4. update the miparAPI environment variable with the location of the miparAPI folder in the new environment

> If you run into any problems during installation, please contact support@mipar.us for assistance

Troubleshooting

Error	During	Cause	Possible Solution
API is not able to find the MATLAB Runtime	Startup	MATLAB Runtime is not installed or is not added to the PATH	1. Check if MATLAB Runtime is installed (default install directory: C:\Program Files\MATLAB\MATLAB Runtime\v911 2. If installed go to step 3, if not installed, download and install from here: https://www.mathworks.com/products/compiler/matlab-runtime.html 3. Check if the following directory was added to the Environmental System Variable 'Path' directory: 'C:\Program Files\MATLAB\MATLAB Runtime\v911\runtime\win64'
API is not able to find the preferences file	Startup	The preferences file parent directory was not added as an environmental variable	Add the environmental variable 'miparAPI' with value site-packages folder in the Python install path. Default: C:\Users\\AppData\Local\Programs\Python\Python[VERSION]\Lib\site-packages\miparAPI
Python returns error: `module 'miparAPI' has no attribute 'initialize'`	Startup	The python library was not properly installed into the environment	1. navigate to the miparAPI library install directory (default C:\Program Files\miparAPI) 2. Try executing the following command from a terminal: `python setup.py install` 3. If 2 fails, navigate to the system install directory (default: C:\Program Files\miparAPI) # copy contents to your python environment 4. Execute the following command from a terminal: `python setup.py install`
`activate` or `activatePK` methods fail and License Manager page is not reachable	Runtime	The license manager was not installed	1. Download the License Manager package for Windows 2. Extract the downloaded **Install_License_Manager_Win.zip** (Right-click > Extract All) 3. Inside the extracted folder, double-click **Install License Manager.bat**

Update Instructions

Major Update (v _.x.x)

1. Download the install pack from provided link
2. Make sure the system is running a Python version that is compatible with the MIPAR Library
3. Run MIPAR-API_vx.x.x_InstallPack.Win.exe and follow the wizard

Minor Update (v x. _ . _) into Standard Environment

1. Download update pack from provided link
2. Make sure the system is running a Python version that is compatible with the MIPAR Library
3. Run miparAPI_v4.1.1.0_Updater_Win.exe and follow the wizard

Minor Update (v x. _ . _) into Custom Environment

1. Download update pack from provided link
2. Make sure the system is running a Python version that is compatible with the MIPAR Library
3. Run miparAPI_v4.1.1.0_Updater_Win.exe and follow the wizard
4. Copy contents of the site-packages/miparAPI folder in the Python install directory to the new environment

System Requirements

Supported Python Versions

- Only versions v3.7, 3.8, 3.9

Supported Operating Systems

Windows (64-bit only)

- Windows 11
- Windows 10
- Windows 8.1
- Windows 7 SP 1
- Windows Server 2016
- Windows Server 2012 R2
- Windows Server 2012

Minimum System Requirements

- **CPU:** Dual-core modern Intel / AMD
- **GPU:** Intel HD 4000 level graphics
- **Storage:** HDD 5400 RPM
- **RAM:** 4 GB DDR2 800 MHz

Recommended System Specs

- **CPU:** 8+ core modern Intel / AMD
- **GPU:** Nvidia Quadro 600 or greater
- **Storage:** SSD SATA 3.0 or greater
- **RAM:** 16+ GB DDR3 1600 MHz

MIPAR API performance greatly depends on workstation hardware. Workstation that is running the API should have similar specifications to the workstation where the recipe was built in order to achieve similar performance.

GPU Support

Enable

Only NVIDIA GPUs are supported for GPU computation at this time. Click here to learn more about the technical aspects and benefits of GPU computation.

To enable GPU computation on the API Machine:

1. Enable GPU computation on the recipe building workstation Link
2. Copy miparPreferences.pref file in C:\Program Files\MIPAR directory of the recipe building machine
3. Paste file into C:\Program Files\MIPAR directory of the API machine

> If you are executing the API on the same machine as recipe building, both applications share the same GPU acceleration settings

Supported Functions

See GPU Computation for detailed list.

Library

This is the current list of methods in the MIPAR API

Method	Purpose	Link
`initialize`	Initialize miparAPI library	Link
`startLicenseMan`	Initialize license manager	Link
`activatePK`	Activate product key	Link
`exeRecipe`	Apply specified recipe to specified image	Link
`setPrefs`	Set preference value	Link
`getPrefs`	Get preference value	Link
`terminate`	Stop miparAPI and background engine	Link

activate()

> ! This function is available in versions 3.2.3 and later. In versions 3.2.2 and earlier, use activatePK() instead.

The `activate` function is used to activate a MIPAR API license key using the command line.

Arguments

```
Args:
productKey: MIPAR API product key or path to a .v2c license file, formatted as a
string.

Returns:
errorStatus: dictionary that contains the errorFlag and errorMessage
flag: 0 - no error, 1 - an error occurred
message: error string of error that occurred during activation
```

Syntax

```
errorStatus = mipar.activate(productKey)
```

Example

```
# initialize engine and licence manager
import miparAPI
mipar = miparAPI.initialize()
productKey = r'12345678-abcd-1234-abcd-123456789101'
errorStatus = mipar.activate(productKey)
# or
productKey = r'directory/apiKey.v2c'
errorStatus = mipar.activate(productKey)
```

aboutAPI

The `aboutAPI` function is used get information about the API version and license status

Arguments

```
Args:
no inputs

Returns:
aboutDictionary with the following keys:
version: API version as a string
active: returns 1 if the machine has an active license
res: returns status from the license manager
```

Syntax

```
aboutDictionary = mipar.aboutAPI()
```

Example

```
# initialize engine and licence manager
import miparAPI
mipar = miparAPI.initialize()
aboutDictionary = mipar.aboutAPI()
mipar.terminate()
```

activatePK()

> ! This function is available through version 3.2.2. In versions 3.2.3 and later, use activate() instead.

The `activatePK` function is used to activate a MIPAR API license key using the command line.

Arguments

```
Args:
productKey: MIPAR API product key formatted as raw string.

Returns:
errorStatus: dictionary that contains the errorFlag and errorMessage
errorFlag: 0 - no error, 1 - an error occurred
errorMessage: error string of error that occurred during activation
```

Syntax

```
errorStatus = mipar.activatePK(productKey)
```

Example

```
# initialize engine and licence manager
import miparAPI
mipar = miparAPI.initialize()
productKey = r'12345678-abcd-1234-abcd-123456789101'
errorStatus = mipar.activatePK(productKey)
```

exeRecipe()

The `exeRecipe` function is used to execute a given recipe on a given image and return the binary mask data and their corresponding measurements.

> ✻ As of 3.4.0, exeRecipe() will no longer call startLicenseMan() before recipe execution, it must be called before exeRecipe. Also, exeRecipe's return dict also contains any messages, warnings, and errors encountered during recipe execution.

Arguments

```
Args:
image: file path to image as a raw string or image matrix. For companion images i
nput a list of raw strings or image matrices. numpy.ndarray image support: numpy
arrays must be converted to a list, see example below.
recipePath: file path to recipe as raw string
cal: calibration factor in units of micrometer per pixel as a float (optional as
of 3.2.3)

Returns:
recOutput: dictionary variable that contains the binary masks of segmentation an
d measurements. Binaries can be retrieved by calling the layer and 'binary' key a
nd measurements are retrieved by calling the layer name and the measurement key.
```

Syntax

```
X = mipar.exeRecipe(imagePath, recipePath) #to keep existing calibration factor s
aved with the recipe
X = mipar.exeRecipe(imagePath, recipePath, cal) #to override the existing calibra
tion factor
```

Examples

```
# initialize engine and license manager
import miparAPI
import cv2
mipar = miparAPI.initialize()
mipar.startLicenseMan(nargout = 0)

# image and recipe paths
```

```
recipePath = r'C:\Users\USER\Documents\Python\ExampleRecipe.rcp'
imagePath = r'C:\Users\USER\Documents\Python\\ExampleImage.png'

# numpy image (alternative to using an image path)
img = cv2.imread(r'C:\Users\USER\Documents\Python\ExampleImage.png)
imagePath = img.tolist()

# setting up python list for passing in companion images (only if your recipe use
s dynamic load companion steps)
originalImage = r'C:\Users\USER\Documents\Python\\ExampleImage.png'
companion1 = r'C:\Users\USER\Documents\Python\\Companion1.png'
companion2 = r'C:\Users\USER\Documents\Python\\Companion2.png'
imagePath  = [originalImage, companion1, companion2]

# execute Recipe
recOutput = mipar.exeRecipe(imagePath, recipePath, 0)

# retrieve dictionary keys
print(list(recOutput.keys())) #layer names
print(list(recOutput['LayerName'].keys())) #measurement names
print(list(recOutput['LayerName']['featMeas'].keys())) #retrieve feature measurem
ent keys

# retrieve data
binaryOutput = recOutput['LayerName']['binary'] #retrieve binary segmentation
measurementOutput = recOutput['LayerName']['MeasName'] #retrieve global measureme
nt
featureMeasurementOutput = recOutput['LayerName']['featMeas']['MeasName']._data
#retrieve feature measurement raw data
calibrationOutput = recOutput['cal']['value'] #retrieve calibration value in unit
s of micrometers per pixel
measurementUnits = recOutput['cal']['units'] #retrieve the measurement units
messages = recOutput['Messages'] #retrieve messages, warnings, and errors from re
cipe execution.  Present in v3.2.3 and later

# MeasNames can be Global Measurements such as: Area, Area Fraction and Number De
nsity.
# It can also be Feature Measurements statistics such as: Mean Caliper Diameter,
Min Roughness, Max Equivalent Diameter or StdDev Area Fraction.
# The notation is concatenated, for example the key for Global Number Density is
NumberDensity and the key for Mean Feature Caliper Diameter is MeanCaliperDiamete
r.
# It is recommended to pull the keys of measurements to get a complete list of me
asurements in the recipe.
```

```
# terminate engine
mipar.terminate()
```

Measurements Units

Image Measurement Class	**Examples**	**Units**
Counts, Intensities, Similarity	Count, Estimate Count, Intensity, Mutual Information, Correlation Coefficient	unitless
Areas	Area	pixels (px) or unit^2 (um^2, in^2, ft^2)
Lengths	Perimeter, Intercepts, Image Dim	pixels (px) or unit (um, in, ft)
Inverse Lengths	Perimeter Fraction, Inverse Intercepts	1/px or 1/unit (um, in, ft)
Densities	Number Density	features/px or features/unit^2 (um^2, in^2, ft^2)
Fractions	Area Fraction, Count Fraction	percent (%)
Feature Measurement Class	**Examples**	**Units**
Counts, Intensities, Shape	Companion ID, Companion Features Intensity, Roughness, Eccentricity, Aspect Ratio	unitless
Areas	Area, Companion Area	pixels (px) or unit^2 (um^2, in^2, ft^2)
Lengths	Diameters, Lengths, Neighbor Distance	px or unit (um, in, ft)
Combination	Perimeter/Area	1/px or 1/unit (um, in, ft)
Location	Centroid	pixels (px) or unit (um, in, ft)
Orientation	Orientation, Tilt	degrees (deg)
Local Measurement Class	**Examples**	**Units**
Counts, Intensities, Shape	Count, Anisotropy	unitless
Fractions	Area Fraction	percent (%)

Shape	Curvature	1/px or 1/unit
Densities	Number Density	features/px or features/unit^2 (um^2, in^2, ft^2)
Lengths	Thickness, Nearest Distance	pixels (px) or unit (um, in, ft)

getPref()

The `getPref` function is used to retrieve the value of a preference in the preference configuration file.

Arguments

```
Args:
key: list of preference keys.

Valid keys:
units

Returns:
value: list of values at keys.
```

Syntax

```
values = mipar.getPref(keys)
```

Examples

```
# initialize engine and licence manager
import miparAPI
mipar = miparAPI.initialize()
mipar.startLicenseMan(nargout = 0)
key_list = ['units']

# get preference values
values = mipar.getPref(key_list)

# terminate engine
mipar.terminate()
```

initialize()

The `initialize` method is used to launch the miparAPI engine and return the engine object. This method must be executed before any other command.

Arguments

```
Args:
None

Returns:
engine object
```

Syntax

```
X = miparAPI.initialize()
```

Examples

```
import miparAPI
mipar = miparAPI.initialize()
```

recipeMeta()

The `recipeMeta` function is used to retrieve meta data from a recipe file.
Currently returns: MIPAR version, calibration factor, units, date and time last modified, layers, layer colors and measurements.

Arguments

```
Args:
recipePath: file path to recipe as raw string

Returns:
recOutput: dictionary variable that contains the metadata of the recipe
```

Syntax

```
X = mipar.recipeMeta(recipePath)
```

Examples

```
recipePath = r'C:\Users\PaulS\Documents\Python\Test_Folder\Training.rcp'
recipeMetadata = mipar.recipeMeta(recipePath)
```

```
lastModified = recipeMetadata.get('last_modified') #date last modified
MIPARVersion = recipeMetadata.get('version')       #MIPAR version
calibration =  recipeMetadata.get('cal')           #calibration
units =  recipeMetadata.get('units')               #units
layers = recipeMetadata.get('layers')              #layers
colors = recipeMetadata.get('colors')              #colors that correspond to th
e layers in order
measurements =  recipeMetadata.get('meas)          #measurements
```

setPref()

The `setPref` function is used to set the value of a preference in the preference configuration file.

Arguments

```
Args:
key: list of preference keys.
value: list of values corresponding to the keys.

Valid keys:
units

Returns:
None
```

Syntax

```
mipar.setPref(keys,values, nargout = 0)
```

Examples

```
# initialize engine and licence manager
import miparAPI
mipar = miparAPI.initialize()
mipar.startLicenseMan(nargout = 0)
key_list = ['units']
value_list = ['um']

# get preference values
values = mipar.setPref(key_list. value_list)

# terminate engine
mipar.terminate()
```

setPyEnv()

The `setPyEnv` function is used to create the preferences file in the miparAPI folder in Python install directory

Arguments

```
Args:
python system path

Valid Path
output of sys.path

Returns:
None
```

Syntax

```
mipar.setPyEnv(sysPath)
```

Examples

```
# initialize engine and licence manager
import miparAPI
import sys
mipar = miparAPI.initialize()

libPath = sys.path
mipar.setPyEnv(libPath, nargout = 0)

# terminate engine
mipar.terminate()
```

startLicenseMan()

The `startLicenseMan` method is used to initialize the license manager. You cannot execute recipes before initializing the licence manager, and it only needs to be executed once per instance of the application.

> As of 3.2.3, it is not necessary to call this method directly, as exeRecipe will do so at the start of recipe execution if necessary. Also, this method will now install the license manager if it is not already present on the system and will restart the license manager service if it is stopped.

Arguments

```
Args:
nargout = 0 to specify that the method is not returning anything

Returns:
None
```

Syntax

```
mipar.startLicenseMan(nargout = 0)
```

Examples

```
import miparAPI
mipar = miparAPI.initialize()
mipar.startLicenseMan(nargout = 0)
```

terminate()

The `terminate` method is used to exit the engine

Arguments

```
Args:
None

Returns:
None
```

Syntax

```
miparAPI.terminate()
```

Examples

```
import miparAPI
mipar = miparAPI.initialize()
mipar.terminate()
```

Release Notes

v3.3.1 API Release Notes

Functionality

- All engine changes have been brought over from: v3.3.0

v3.3.4 API Release Notes

Functionality

- All engine changes have been brought over from: v3.3.4, v3.3.3, and v3.3.2

v3.4.0 API Release Notes

Functionality

- All engine changes have been brought over from: v3.4.0
- exeRecipe now accepts a python numpy array as an input: exeRecipe
- added aboutAPI method which returns the API version, and license status: aboutAPI

Hotfixes

3.4.0.2

- Summary stats of some local measurements would not report correctly
- For some local measurements, CSV files from a batch process would contain the identical data for all images
- Headers in CSV files of batch local measurements had issue with units formatting

3.4.0.1

No changes. Version change done only to keep up with Base product.

v3.4.1 API Release Notes

Functionality

- All engine changes have been brought over from: v3.4.1
- Measurement keys no longer contain the units and the spelling was corrected.

v3.4.2 API Release Notes

Functionality

- All engine changes have been brought over from: v3.4.2

v3.4.4 API Release Notes

Functionality

- setPref: this method can now be used to update the GPU device selection by passing in the ‘gpuDeviceChoice’ key with an integer starting with 1 for the first GPU device on the system.
- All engine changes have been brought over from: v3.4.4

v4.0.0 API Release Notes

Notes

- All engine changes have been brought over from: v4.0.0
- Support for Windows 11 systems
- Support for Python 3.7, 3.8 and 3.9. Note API no longer supports Python 3.6

v4.0.1 API Release Notes

Notes

- All engine changes have been brought over from: v4.0.1

v4.1.0 API Release Notes

Notes

- All engine changes have been brought over from: v4.1.0

v4.1.1 API Release Notes

Notes

- All engine changes have been brought over from: v4.1.1

REST API

Welcome to MIPAR REST API documentation. This documentation will describe the library capabilities and outline some example workflows.

> ! The REST API is still in Beta and under active development. Updates will be posted in the Release Notes

Topics

Introduction

MIPAR REST API is a powerful tool that packages the MIPAR engine into a Docker Image

Scope
The MIPAR REST API was developed to meet the needs of customers who are already using MIPAR to do powerful image analysis, but would like to setup “headless” workstations that execute algorithms with minimal supervision. The REST API is deployed using Docker as it is extremely versatile and widely adopted.

Capabilities
At this time the MIPAR REST API is only able to execute pre-built MIPAR recipes, this means that you must have an already developed MIPAR recipe to utilize the MIPAR REST API. The Docker image accepts images, recipes and returns the resulting binary masks and global measurements.

System Requirements

Minimum System Requirements (Docker)

Docker for Windows

Recommended System Requirements

- **CPU:** 8+ core modern Intel / AMD
- **GPU:** Nvidia Quadro 600 or greater
- **Storage:** SSD SATA 3.0 or greater
- **RAM:** 16+ GB DDR3 1600 MHz

Install and Activate

Standard Image Install

1. Docker is a prerequisite to install the REST API, you can download docker here: LINK
2. MIPAR support team will request the workstation Windows Edition and Version to ensure the Docker Image is compatible
3. Make sure Docker is running
4. Run the following command to log into the repository: `docker login --username AWS --password <TOKEN> 406796911473.dkr.ecr.us-east-1.amazonaws.com`
5. The `<TOKEM>` will be provided by the MIPAR support team. Note the token is only valid for 12 hours.
6. Next run `docker pull <AWS RESOURCE NAME>` to pull the Docker image to the system
7. The `<AWS RESOURCE NAME>` is specific to the user's Windows Version and will be provided by the MIPAR support team.

Install License Manager

1. Download the License Manager package for Windows
2. Extract the downloaded **Install_License_Manager_Win.zip** (Right-click > Extract All)
3. Inside the extracted folder, double-click **Install License Manager.bat**

Add Exception to Windows Firewall

This allows the Docker image to communicate with the license manager on the host machine

1. Windows Defender Firewall>Advanced Settings
2. Inbound Rules>New Rule
3. Program: C:\Program Files (x86)\Common Files\Aladdin Shared\HASP\halplms.exe
4. Allow the Connection
5. Rule should apply based on internal security settings

Activate Trial License

1. Activate trial license.v2c file on the API system Link

Activate License

See Activate POST command: Link

POST Method

There are two commands that can be sent to the REST API using a POST call: `process` and `activate`

Start the MIPAR REST API Docker image

- With docker hub running, execute the command `docker run -it -p <PORT> <IMAGE>`
- The `<PORT>` can be set by the user, for example `8000:8000` and the `<IMAGE>` is specific to the API version and Windows Edition, for example `mipar-api:1.0.4.w.2004`

Activate command

The `activate` command is used to activate a product key, note: trial key file (.V2C) can be applied using instructions here: LINK
Format the HTTP `activate` command:

```
POST /activation HTTP/1.1
Host: localhost:8000
Content-Type: application/json
Content-Length: 71
{ "ip":<HOST IP>, "pk": <PRODUCT KEY>}
```

Format HTTP CURL `activate` command:

```
curl --location --request POST 'http://<HOST IP>:8000/activation' \
--header 'Content-Type: application/json' \
--data-raw '{ "ip":<HOST IP>, "pk": <PRODUCT KEY> }'
```

Activate command output

Returns the following JSON structure:

```
{
    "flag": <STATUS>,
    "message": [<MESSAGE>]
}
```

`<STATUS>` is 0 when the license was activated successfully and 1 when it failed
`<MESSAGE>` is empty when the license was activated successfully and an error string when activation failed

Process command

The `process` command is used to apply a specified recipe to a specified image.
`binary` key can be set with a value of `true` or `false` to enable or disable the binary image output
Format HTTP CURL `process` command:

```
curl --location --request POST 'http://localhost:8000/process' \
--form 'image=@"/path/to/file"' \
--form 'recipe=@"/path/to/file"'
--form 'binary = "true"'
```

Process command output

See JSON Output: LINK

JSON Output

Process JSON Structure Output

The `process` call returns a JSON structure that represents the recipe output. Field explanation:

```
<LAYER NAME>: any layers that were added to the recipe
binary: the logical image that shows what was detected
<MEASUREMENT STAT>: statistic of any measurement that was added to the recipe. Ex
ample: "MeanCaliperDiameter" or "StdDevCaliperDiameter"
<MEASUREMENT_1 FEATURE MEASUREMENTS>: measurement of individual features in the d
etected image Example: "CaliperDiameter" or "MinimumDiameter"
scaleFactor: returns the scale factor in units micrometers/pixel
measurementUnits: returns the base unit of the measurement. Example: base unit: m
m, diameters: mm. areas: mm^2, number density: features/mm^2.
Messages: contains errors, warning and messages that the recipe returned during e
xecution
```

For complete table of measurement units see: LINK

Example Output

```
{
    <LAYER NAME>: {
        "binary": [
            [
                <LOGICAL ARRAY OF ARRAYS>
            ]
        ],
        <MEASUREMENT STAT_1>: double
        <MEASUREMENT STAT_2>: double
        "featMeas": {
            <MEASUREMENT_1 FEATURE MEASUREMENTS>: [
                [
                   <DOUBLE ARRAY>
                ]
            ]
        }
    },
    "scaleFactor": double
    "measurementUnits": string
    "Messages": {
```

```
        "errors": [],
        "warnings": [],
        "messages": []
    }
}
```

Error Code Troubleshooting

Error	Cause	Possible Solution
STATUS 500	• Docker container is not running • Docker container is not reachable	• Check if the Docker container is running • Check if the Docker is reachable by making a call on the host machine using a too like Postman • Check if the port is open in the firewall
STATUS 200 with a non empty error key	Something went wrong with recipe execution	Test recipe execution using the MIPAR GUI application
STATUS 400 with a license error	• API was not activated on this machine • Docker container is not able to reach the license manager on the host machine	• Check if the API was activated on the host machine by checking this page: http://localhost:1947/_int_/features.html for API feature • Add hasplms.exe to the Windows Firewall inbound Rule

Release Notes

v1.1.0 Release Notes

Bug Fixes

- Improved error handling when API failed to communicate with the license manager
- Improved warning handling when MIPAR recipe is out of sync with MIPAR version
- JSON output follows standard protocol for ‘NAN’ and ‘inf’ values
- Interruptible recipe steps and out of range crop steps are now properly handled and no longer stop recipe execution

Release Notes

v1.x.x Release Notes

v2.x.x Release Notes

v3.0.0 Release Notes

Major New Features

Deep Learning

The most significant new technology to come to MIPAR yet. Teach MIPAR to detect even the most complex features simply by tracing. Models are trained in the new Deep Learning Trainer. Trained models can be applied to images as part of a Recipe in the Image Processor.

The ability to apply models as part of a Recipe is included with the Base Package. Access to the Deep Learning Trainer to train your own models requires the Deep Learning Extension.

Learn more about tracing features.
Learn more about training models.
Learn more about applying models

See examples of some of the breakthrough solutions Deep Learning has enabled.

Report Generator

Generate formal Word Doc and PDF reports which include measurements, reference and segmented images, and experiment data. Available report types are below. Learn more here.

Report Type	Generated From
Single Image Global	Global measurement table in Image Processor
Single Image Feature	Color by Measurements
Single Image Local	Local Measurements
Batch Image Global	Global measurement table in Post Processor
Batch Image Feature	Histogram of Measurements opened by Post Processor

Portable Sessions

Save Session files that can be opened from other locations or computers. SSN2 file extension indicates portable Session file. Checkbox in BP and RTP will toggle portability. SSN2 files can also be saved from PP and DLT.

Quality-of-Life Improvements

Keyboard Shortcuts

Many added keyboard shortcuts, especially in windows with manual editing tools. Several shortcuts made universal across all windows. Learn more here.

Sortable Tables

Click on column headings in any table to sort by that column. Click again to toggle ascending vs. descending order. Sorting caries over to reports made with the Report Generator.

Go-to-Image in PP

Generate a measurement table in the Post Processor, and double-click on a cell to show the corresponding image. Useful for finding an image with features at measurement extremes.

Save in More Formats

Save images in formats including PNG, JPG, BMP, GIF, and HDF4 across all apps.

Open Pyramid TIFs

Pyramid TIF files now supported. The highest resolution plane will open.

Move Layer

Move a Layer from one step to another without removing and re-adding the Layer. Select both steps and the “Set Layer” button changes to “Move Layer”.

CSV Default Output

All spreadsheets now export in CSV format by default, allowing for easier 1-click opening in Excel..

Fewer Save Dialogs

“Save Complete” dialogs have been moved to transient status bar messages.

Cleaner Batch Output

All outputs from a batch process are now placed in a folder with the same name as the Session file.

3D Toolbox Improvements

Cleaner axes tick marks. Improved surface and volume exporting. More consistent perspectives between 2D and 3D viewers. More precision in principal axes output by *Aspect Ratio* feature measurement.

Functions

Enter Dimensions in Crop Image

Manually enter or adjust crop dimensions directly from *Edit > Crop Image*.

Replace with Local Normal

New *Clean-Up > Replace With > Local Normal* function to replace each pixel in a sparse line with a short line perpendicular to the line. Useful for draw thickness measurement line probes to allow for batch measurement of local thickness.

Pixel Lines

New *Morphology > Pixel Lines* function for ensuring all pixels in 1-pixel thick lines are connected at their faces.

Percentage Mode in Basic Threshold

New mode in *Segmentation > Basic Threshold* for selecting pixels which are above or below a certain percentage of the grayscale space. Useful for selecting features from probability maps produced by *Deep Learning > Apply Model* (e.g., set “Percentage” to 90 in “Bright” mode to select pixels which are at least 90% likely to belong to features of interest).

Choose Color in Manual Edit

Change the overlay or outline color in *Segmentation > Manual Edit*. Selected color is saved into the Recipe step.

Measurements

Measure Tool in PP

Interactive point-and-click Measure Tool (ruler icon) now available in the Post Processor.

First Moment of Area

The feature measurement *First Moment of Inertia* has been replaced by *First Moment of Area* (*First Moment of Volume* in 3D Toolbox), which is a scale invariant metric of how much feature area (or volume) extends away from its centroid. See the description here for more details.

v3.0.1 Release Notes

Bug Fixes

Deep Learning Training

Fixed an issue where some deep learning models would not train based on their layer names.

Fixed an issue where some deep learning models would not train from a Session output from a batch process.

Portable Session Saving

Fixed an issue where portable Sessions would save as a .ssn file during overwriting.

Replace Image Sorting

Fixed an issue where “Sort on Load” setting in PP and DLT was not respected when bulk replacing images.

v3.0.2 Release Notes

Improvements

Deep Learning Training

Tiles without any labeled pixels will be ignored during training. This can dramatically speed-up training of some models.

All error dialogs during training now contain more descriptive messages.

Bug Fixes

Deep Learning Training

Fixed an issue where tiling was not respected if one of the tiles was set to 1.

Fixed an issue where some errors did not properly print to the log file.

Fixed an issue where GPU memory overload during application was not handled properly.

First Moment of Area Units

Fixed an issue where First Moment of Area units were printed incorrectly.

v3.0.3 Release Notes

Bug Fixes

Manual Edit

Fixed an issue where *Manual Edit* step failed to open during edit if “Labels” was the default viewing mode.

Recipe Saving

Fixed an issue where if you had a Recipe with only measurement steps you would not one prompted to save the Recipe upon opening a new image.

Erase All Layers

Fixed an issue where “Erase All Layers” option would not work properly in the Post Processor under certain conditions.

v3.0.4 Release Notes

Bug Fixes

- Fixed bug where MIPAR loads a color image as grayscale due to small thumbnail in second TIF page
- Fixed bug where error tone is given when entering a factor in *Resize Image*
- Fixed bug where table from “Ruler Tool” is not shown properly in Post Processor
- Fixed bug where “Save All Images” would occasionally fail in Post Processor and Deep Learning Trainer
- Fixed bug where “Number Density” measurement wouldn’t clear after reloading session in Post Processor
- Fixed bug where Layer legend in batch feature measurement reports wouldn’t respect the Session layer colors
- Fixed bug where *Save All Layers* would occasionally set Recipe Panel invisible in Simple Mode
- Fixed bug with comma delimiting when saving all intercepts from an *Intercepts* measurement
- Fixed bug where *Apply Model* would occasionally improperly trigger an error about wrong image type
- Fixed bug where Recipe loading would not cancel properly when choosing not to resize mismatched Companion Images
- Fixed bug where some measurements could be left out of a batch global measurement report

Improvements

- Fixed issue where error message windows can move behind Post Processor and Deep Learning Trainer
- Fixed issue where close confirmation box would appear behind *Pattern Mapping* window
- Fixed issue where clicking “Open” in Deep Learning Trainer would not ask to clear or reload active Recipe
- Image Processor will now warn you when saving Recipes with disabled steps
- File picker now uses default .CSV format for loading measurements in *Color By Measure*, *Histogram of Measurements*, and *Local Measurements*
- Clarified message regarding expired Maintenance Plan
- Streamlined tips shown in Image Processor

v3.0.5 Release Notes

Bug Fixes

- Addressed bug where resizing *Manual Edit* windows can cause editing tools to crash
- Fixed bug where *Manual Edit* would crash when “interruptible” while running a batch which contained a *Crop Image* step
- Fixed bug where white background would not show when hiding the first layer in Layers View with 100% opacity
- Fixed bug with opening images in Post Processor by double-clicking a cell in a global measurements table
- Fixed bug with removing multiple slices from an Image Stack in 3D Toolbox

Improvements

- Recipes now always load in Simple Mode if they contain Layers

v3.1.0 Release Notes

New Features

Copy & Paste

It’s finally here. Now you can copy and paste an unlimited number of Recipe steps.

To copy: click and drag, Shift-click, or Ctrl-click to select multiple steps and right-click > “Copy”, or “Ctrl+C” (“Cmd+C” on Mac). To paste: select where in the Recipe to paste and right-click > “Paste”, or “Ctrl+V” (“Cmd+V” on Mac).

The clipboard is persistent across your system, so you may copy & paste in the following workflows:

- **In Same Recipe:** Copy from Recipe > Paste in same Recipe
- **Between Recipes:** Copy from Recipe > clear Recipe > load new Recipe > Paste
- **Between Two Open Recipes:** Copy from Recipe in one MIPAR instance > Paste in Recipe in second MIPAR instance

Your last copied steps will even be available to paste after a system restart.

Batch Feature Measurements

With a checkbox in BP and RTP, you can now choose to output all individual feature measurements during a batch process.

Feature measurement spreadsheets for each image will appear in folders organized by Layer in “[Session Name] > measurements > [Layer Name]”.

Merge Sessions

Now you can combine multiple Sessions into a single file. Useful especially for adding new batch process results to an existing deep learning Session, in order to re-train an improved model.

Any Session type (Portable or Regular) can be merged into any other Session type. Both Sessions must have the same number of Layers.

To merge, open a Session in PP or DLT, choose “Merge Session” from the shortcut bar, and choose the Session to merge into the current.

Edit Other Layers

A new dropdown in the lower left corner of the manual editing panel in PP and DLT allows more flexibility to perform edits in other Layers.

In Fill Mode, choose between “Fill This Layer” “Fill All Layers”, and “Erase Other Layers”.

In Erase Mode, choose between “Erase This Layer” “Erase All Layers”, and “Fill Other Layers”.

New Measurements

The following feature measurements have been added:

- **Minimum Diameter:** Smallest line length that fits between two parallel lines tangential to each feature.

- **Tilt**: Angle of the caliper diameter of each feature with respect to the positive X-axis. Positive angles are clockwise rotations and negative counterclockwise.

Improvements

- Bio-Format images can now be used to train deep learning models in the DLT
- Bio-Format images can now be batch processed into 3D Reconstructions in the BP and RTP
- Layer folders are now renamed in portable Sessions when Layers are renamed and the Session is saved
- Deep learning models can be dragged directly into the IP to auto-launch *Apply Model* and auto-apply the model
- Deep learning models can be dragged into *Apply Model* to load the model

Bug Fixes

- Fixed bug with Overlay and Labels viewing mode of blank images in PP and DLT
- Fixed bug with color picker in *Manual Edit*
- Fixed bug where editing a hidden Layer in the IP would cause it to un-hide

v3.1.1 Release Notes

Bug Fixes

- Fixed bug where *Feature Measurement* steps would fail for Layers with no features

See new features released in v3.1.0

v3.1.2 Release Notes

Bug Fixes

- Fixed bug where training settings panel in DLT could not be visible
- Fixed bug where “Remove Chapter” option in IP could not enable
- Fixed bug with “Undo” in PP and DLT

Improvements

- Progress indicator now appears in 3DTB when counting features when labels were not up-to-date

See new features released in v3.1.0

v3.2.0 Release Notes

New Features

Choose Format When Saving Multiple Images

You can now choose between the following formats when using “Save All Images” from the Post Processor and Deep Learning Trainer:

- TIFF
- PNG
- JPEG
- BMP
- HDF4

Re-designed Pattern Mapping

Pattern Mapping has been re-designed and dramatically simplified to output a grayscale map, which can then be further processed to the final segmentation.

Integrated Product Key Activation

If your system is online, you can now activate your Product Key for a local license with a single click.

Streamlined Installation

For Windows installations, the MIPAR Installer and Setup Assistant are now executed automatically after doubling clicking the InstallPack.

Improvements

- *Gaussian Blur* can use sigma values under 3
- *Gradient Filter* can use sigma values under 3
- *Find Texture* uses faster hessian matrix calculation
- *Perimeter Pixels* now produces an inverted, more intuitive result. Recipes with this step will still perform under the old operation until resaved in v3.2.0.
- A new chapter point will no longer be automatically set when calling a selection (binary) Memory Image
- When calibrating the scale factor in the Post Processor, you may now choose from any of the loaded reference images
- You will now be warned in the 3D Toolbox when generating feature measurements from labels which

are not up-to-date

Bug Fixes

- Fixed bug in the Post Processor where opening a reference image did not trigger a “Clear or Reload” prompt in the Image Processor
- Fixed bug where features that exist only on the perimeter of the image would have their caliper diameters measured as 1
- Fixed bug in the Image Processor where copy/paste of text would not work in the help/notes or scale factor boxes
- Fixed bug in 3D Toolbox where tick labels in the 3D Viewer would not appear properly when measurement units were switched from microns
- Fixed bug in 3D Toolbox where the bright/dark selection was not respected in *Basic Threshold* > *3D*

v3.2.1 Release Notes

Bug Fixes

- Fixed bug where recipes would not run properly when making Minimum Diameter feature measurements on layers without features
- Fixed bug with launching Save All Images when reference images had .jpeg or .tiff extensions
- Fixed bug where Report Generator settings were at times not saved

See new features released in v3.2.0

v3.2.2 Release Notes

New Functions

- *Replace With > Nearest Distance* replaces each feature with a line between it and its nearest neighbor.

Bug Fixes

- Fixed bug where inserting an *Apply Model* step below another doesn't update the dependencies of later *Call Output* steps
- Fixed bug where pasting several steps above step 1
- Fixed bug in *Pattern Mapping* when using FFT Matching mode
- Fixed bug with removing a step above *Register Image*
- Fixed bug with running *Replace With* > *Caliper Diameter* and *Replace With* > *Minimum Diameter* steps
- Fixed bug with Post Processor remembering the "Fill/Erase in Other/All" dropdown when switching between Fill and Erase modes
- Fixed bug with training deep learning models with periods in Layer names

Improvements

- Turning off Hardware Acceleration in Preferences now disables additional acceleration to workaround potential issues in Windows 7
- Pixel units for area measurements now px^2, and Perimeter Fraction now 1/px

See new features released in v3.2.0

v3.2.3 Release Notes

New Functions

- *Crop Image > Apply Last* crops the image using the area defined in the last Crop Image step in the recipe.

Bug Fixes

- Fixed bug with global measurement report printing
- Fixed bug with applying pre-loaded model from Deep Learning Trainer
- Fixed bug where Deep Learning Trainer did not respect GPU selection preference
- Fixed bug with certain workflows for flagging recipe steps and setting notes
- Fixed bug where setting scale factor could cause errors during recipe update when first step was *Crop Image*
- Fixed bug with adding new layers after previous layers were cleared
- Fixed bug where error messages could appear behind *Apply Model* window
- Fixed bug where *Pattern Mapping* would not run on a binary image during recipe update

Improvements

- In *Reject Features* “Proximity Awareness”, Local Window settings now include the feature potentially being rejected
- Faster preference reading
- Removing all recipe steps now behaves identical to pressing “Clear” (trash can)
- Future auto-updates will now use AWS-hosted links
- Posted workaround for adding custom logos into report templates

API

- Individual feature measurements are now output
- Images can now be input as numpy arrays in addition to filenames
- *Load Companion* images can now be input separately, allowing for more powerful batch processing solutions
- Installer has been streamlined
- Error and warning reporting has been improved

See new features released in v3.2.0

v3.2.4 Release Notes

Bug Fixes

- Fixed bug with deleting some steps which depend on others
- Fixed bug where layers could all become temporarily identical when making “Outline” edits in PP and DLT
- Fixed bug where feature measurement stats CSV files were not properly comma separated
- Fixed bug where “Add Layer” button disables after removing all layers in DLT
- Fixed bug where toggling “Show/Hide” in Labels view causes distorted axes in PP and DLT
- Fixed bug in AOI Tracking where “Enter Crop” dialog is hidden behind window
- Fixed bug with removing some *Register Image* steps
- Fixed bug in Preferences with turning off GPU Computation
- Fixed bug where “Show Scalebar” panel would sometimes appear behind others in PP
- Fixed bug with switching from “Manual” to “Memory Image” in *Auto Segmentation*

Improvements

- Sessions can be drag-and-dropped to load in PP and DLT even if one is already open
- Improved performance in all preview windows with continuous sliders
- Standardized number of significant figures in global and feature measurement reports
- Improved filename sorting
- Feature measurements now return NaN (not-a-number) when no features are present
- Icon appears next to “Help” menu when update is available
- Improved error messages when loading non-B/W images into B/W panel in PP and DLT
- *Replace With* > *Triangulation* now works with only two features present
- DLT will now check for minimum number of layers before saving training layers and tiling

[See new features released in v3.2.0]

v3.3.0 Release Notes

New Features

Express Licenses

Express Licenses are now available. These “runner licenses” are available at reduced rates and only allow recipe running (not building). Express Licenses provide all other Base License functionality.

They are an ideal fit for organizations with R&D and production facilities, letting them build/optimize recipes with MIPAR Base, and deploy recipes to production with MIPAR Express. Please contact sales@mipar.us if interested.

Dark Mode

Dark Mode has come to MIPAR! Upon launching the Image Processor after updating, you’ll be prompted to test and choose a theme. Theme can always be changed in File > Preferences from any app.

All-new GPU Acceleration

We’ve rewritten the GPU Computation engine from the ground up. Most recipes will now see significant acceleration when run on compatible GPUs. Nearly all functions now support GPU Computation. See here for more information.

> ! Some functions may produce slightly different results when run on the GPU vs. the CPU.

Improved Editing Performance in PP and DLT

Overall image display performance has been improved in the Post Processor and Deep Learning Trainer, which results in a more responsive manual editing experience.

Higher Resolution Saving

Graphics saved from tools such as Color by Measurements are now saved at higher resolutions. These include graphics such as histograms and images with colorbars.

Powerful FFT Processing

A pair of new functions now allows for custom FFT filters to be crafted through a limitless number of recipe step combinations.

Use *Make FFT* to produce an FFT of the Current Image to use as a canvas for developing your filter. Treat this FFT as you would any other grayscale image and use steps like Basic Threshold, Reject Features, etc. to automate the creation of powerful custom filters.

Set the filter as Companion Image and use *Apply Stored FFT Filter* to apply this filter to an image.

New *Reject Features* Measurements

“Number of Features” rejects based on the number of features in the companion image contained within the feature.

“Number of Holes” rejects based on the number of holes contained within the feature.

New *Load Companion* Setup

Load Companion Image uses a new setup window, which allows for a more streamlined adding and editing experience.

New Measurements

Count Fraction measures the count of features in the Current Image as a percentage relative to the count of features in a chosen Memory Image.

“Intensity Sum” has been added as a new *Feature Measurement* and measures the sum grayscale intensity in the latest Companion Image within each feature in the Current Image.

“Median” is now an available summary statistic for all *Feature Measurements*.

Show/Hide in Image Processor

A “Show/Hide” button has been added next to the Viewing Mode dropdown in the Image Processor. This allows for one-click toggling between the current segmentation and the Reference Image.

Improvements & Minor Changes

- *Smooth Features* can now use “Window Size” values greater than 15
- *Local Threshold* now has a progress indicator

- Export 3D stacks to movies has been optimized to eliminate out-of-memory errors when working with large image stacks

Bug Fixes

> ! **Critical Fix:** Fixed bug which could cause occasional MIPAR hang-ups, where a force restart was the only solution. On Windows systems, Setup Assistant now sets a system environment variable which changes where some cache files are stored. If you have experienced this bug on Windows, we recommend you perform a fresh install here.

- Fixed bug in *Color Cluster* when interacting with image during computation
- Fixed bug in *Manual Edit* when editing the step on a larger reference image than it was developed for
- Fixed bug in *Uniform Dilation* and *Uniform Erosion* where values above the slider max could not be entered
- Fixed bug that caused *Replace with Caliper Diameter* and *Replace with Minimum Diameter* to occasionally fail for very small features
- Fixed display bug in *Highlight Lines* when operating on a B/W image
- Fixed bug where some feature measurement files output during batch would overwrite each other
- Fixed bug with some stats outputs in formal local measurement reports
- Fixed bug with appearance of formal batch feature measurement reports
- Fixed bug when generating measurements in PP where the layer matrix would reset after each generation
- Fixed bug where viewing modes were not properly listed after clearing a PP session and loading B/W images

v3.3.1 Release Notes

Bug Fixes

- [All Apps] Would not be properly informed of unsupported image formats when trying to open
- [Color by Measurements] Changing the format of feature ID labels would duplicate the labels
- [Deep Learning Trainer] Image would not display properly after edits when in a viewing mode other than “Layers”
- [Feature Measurements] Intensity measurements did not report properly
- [Image Processor] Could experience a lag on Mac systems when showing/hiding Measurements Panel and switching recipe modes
- [Post Processor] Could not change the the labels colormap

See new features released in v3.3.0

v3.3.2 Release Notes

New Features

Slice-by-Slice Measurements in 3D

Slice-by-Slice Measurements can now be made in the 3D Toolbox. This enables any of the Feature Measurements to be made from all X-Y slices, where each slice's feature labeling is based on the full 3D labeling. This is useful for monitors features' parameters over the Z-dimension, such as in particle tracking.

Clean Boundaries in 3D

Apply *Clean Boundaries* both slice-by-slice and in 3D in the 3D Toolbox to help remove artifacts between neighboring features.

Improvements

- Significantly improved performance for "Based on Companion" measurements in 2D and 3D. Users can expect 10×-300x acceleration when making measurements such as "count features within other features", "measure grayscale intensity within features", etc.
- Feature IDs stay under scalebar in Color by Measurements
- Layer name now appears next to image name if shown in Color by Measurements and Histogram of Measurements
- Viewing mode controls now disabled when "Hide" is toggled in IP, PP, and DLT
- Removed 'Shift' shortcut for "Show/Hide" in IP
- Tools menu with Histogram of Measurements added to BP, RTP, and DLT
- Feature IDs now save into "Low-Res with Colorbar" image if shown in Color By Measurements

Bug Fixes

- Fixed bug where unsupported file would still appear in list after attempting to add in PP and DLT
- Fixed bug where shift+clicking to zoom out in IP would toggle "Show/Hide"
- Fixed bug with displaying 3D slices in some cases after reconstruction on the GPU

See new features released in v3.3.1

See new features released in v3.3.0

v3.3.3 Release Notes

Bug Fixes

- Fixed bug where feature measurements in PP could in some cases be formatted with incorrect units
- Fixed bug where scale factor would not display correctly in PP when reading a Bio-Formats image with embedded scale factor
- Fixed bug where mouse movement in the image axis could interrupt Preferences launching from PP and DLT
- Fixed bug with dropdown next to “Add Layer” in DLT
- Fixed bug with saving portable Sessions out of DLT
- Fixed bug with removing top image from list in DLT

See new features released in v3.3.2

See new features released in v3.3.1

See new features released in v3.3.0

v3.3.4 Release Notes

Improvements

- *Translate Image* now supports either black or white image padding
- API product will now appear in list after activation if on user's license

Bug Fixes

- Fixed bug which could cause Local Measurements to not launch properly
- Fixed bug where nonexistent directory would not be properly caught when trying to open a recent image

See new features released in v3.3.3

See new features released in v3.3.2

See new features released in v3.3.1

See new features released in v3.3.0

v3.4.0 Release Notes

New Features

Local Measurements in Recipe

> Add local measurements as a step in the recipe! They can be displayed with a simple click after recipe loading and summary stats (e.g., mean, min, max, etc.) of the local measurement will appear in the global measurement report.
>
> To add to the recipe, open *Local Measure* > setup and generate > customize formatting as desired > click “Add to Recipe” in upper-right corner. Full data tables of all local measurements can be exported per image as CSVs during a batch process, just as feature measurements can be today.

Dynamic Scale Factor

> Setup your recipe to dynamically determine the scale factor based on the size of a recognized feature.
>
> In Calibrate Scale, you can now setup a third mode which accepts a “sub-recipe” for detection and a defined dimension to measure. When the main recipe is run, this sub-recipe first detects the scale-determining feature and sets the scale factor according to the feature’s measured pixel and defined physical dimensions.

Dilation and Erosion in Physical Units

> *Uniform Dilation* and *Uniform Erosion* depths can now be defined in physical units such as microns, rather than just pixels. Useful when defining custom regions-of-interest (e.g., border thickness) in physical units, expecially when the recipe needs to accomodate various magnifications.

Companion Area Fraction Measurements

> Area fraction of “child” features, relative to their “parents” can now be made automatically as part of the “Companion Area” measurement under *Measure Features*.
>
> To setup, store the “child” features in the Companion Image, and set the “parent” features as the layer to measure. Choose *Measure Features* > select layer > check “Companion Area” > add to recipe.

Improvements & Minor Changes

- Load large session from Batch Processor has more responsive loading spinner
- Improved file sorting in Batch Processor, Post Processor, and Deep Learning Trainer
- Up/down arrows in Post Processor and Deep Learning trainer can no longer produce mismatched reference and selection image sizes
- *Load Companion Image* now offers an option to force resize if necessary
- *Local Measure* will now open immediately when only one layer in the recipe
- Improved dark mode appearance in some preview windows

Bug Fixes

- Reject Features – Area Fraction (CRITICAL)

  > Features to be rejected using Area Fraction would measure incorrectly. The bug fix corrects the value in the Reject Features step to maintain detection prior to version 3.4.0.
  >
  > Recommended protocol:
  > 1. Load Recipe
  > 2. Edit the Reject Features step that uses Area Fraction
  > 3. Change the value to the appropriate value to meet detection
  > 4. Select ‘Accept’
  > 5. Save Recipe

- Removing layers could cause an improper layer assignment to *Feature Measurement* steps
- MIPAR could fail to launch with certain outdated GPU drivers
- View > Manage Layers would cause *Feature Measurement* steps to reset to “All”
- GPU acceleration could cause freezing
- *Load Companion Image* would fail with a certain image type
- Intermittent issue with replacing reference images in Deep Learning Trainer
- Displayed scale factor units in some generated reports would not match the displayed numerical value
- Detection results from some deep learning model applications could appear offset from the reference image
- “Out of GPU memory” errors now caught in *Adaptive Threshold*
- *Orthogonal Correlate* would produce anomalous results in some cases
- Erroneous reference image replacement in Deep Learning Trainer and Post Processor will no longer break the Session file
- Reference image order could reset after first up/down move in Deep Learning Trainer
- Adding an inverse layer in Deep Learning Trainer would not always appear until the active image was

changed
- In Post Processor an error could occur when opening a batch of results over the top of a previous batch
- Bug with Shift+clicking multiple images in Batch Processor
- Deleting a layer in Deep Learning Trainer or Post Processor would not enable the “Save Session” button
- *Crop Image* would not be applied to reference image in some cases upon loading recipe
- In *Local Measure* “Save with Colorbar” could fail
- In *Local Measure* “Intensity StdDev” measurement could fail
- Viewing mode dropdown no longer active in Simple Mode
- Global intensity measurements in Image Processor would be disabled for selection images that were converted to grayscale

Hotfixes

3.4.0.2

- Summary stats of some local measurements would not report correctly from the IP and BP
- For some local measurements, CSV files from a batch process would contain the identical data for all images
- RTP did not output batch local measurements properly
- RTP could experience an issue and fail to run
- Headers in CSV files of batch local measurements had issue with units formatting
- “Output Local/Feature Meas” checkbox in BP and RTP would not re-enable after batch process finished
- “Auto” mode in Calibrate Scale could fail when search area extended beyond image frame
- DL models saved out of *Apply Model* could have issues being applied in future recipes

3.4.0.1

- Manual distance tool did not disable when using “Auto” mode

v3.4.1 Release Notes

Improvements

- Added a preference to increase the amount of memory available for opening specialty file formats with Bio-Formats
- Removed “Enter Dimensions” entry from Crop menu in 3D Toolbox since functionality is present in Crop > Draw

Bug Fixes

- More specialty file formats now accepted if Bio-Formats library is present
- Proper validation of image files of specialty types
- Anisotropy local measurement raw data would not output properly during a batch process if “Output Local Meas” is checked
- When adding both rotating intercepts and random intercepts measurement, only the last measurement would be printed
- Feature Measurement layer assignment would switch to “All” after deleting a layer step
- Animation tool in 3D Toolbox did not function properly
- 3D Toolbox slice by slice dilation and erosion errored with GPU Acceleration turned on
- 3D Toolbox uniform 3D dilation would error on some Windows machines
- Local Measurements in recipe would appear as columns from Choose Layers in IP
- Line Tool in Image Processor would occasionally not produce a line profile visualization
- Local Measure histogram coloring did not change with changing limits on the colormap
- CSV file headings of Local Measurements were not spelled properly

[See new features released in v3.4.0]

v3.4.2 Release Notes

Improvements

- BP and RTP will now save intermediate session files during batch, such that sessions can be reviewed and/or merged followed a batch cancellation
- In Simple Mode in the IP, “Measurements” is now written beside the Measurements Panel expansion arrow to clarify its function
- Added Feature Number column to Local Measurements that assigns feature numbers to each pixel based on the feature numbering in the layer.
- Added preference to link the default output folder of user saved files to the last opened image.
- When replacing a model in a recipe, calls to the previous model’s output will now track the class names of the new model
- Description text above progress bar now consistently centered
- Error dialogs are now also printed to the log file and the log file no longer gets cleared when MIPAR opens
- Enabled application of recipes designed on RGB images to 3-channel grayscale images (affects *Channel Operation*, *Color Cluster*, *Color Select*, and *Apply Model* on color original)
- *Channel Operation* now uses floating point arithmetic
- Improved handling of GPU out of memory condition with Pre-Processing steps
- Added tooltip text to buttons that can be activated by keyboard shortcuts

Bug Fixes

- File > Save All Layers in PP and DLT errored when using GPU Acceleration
- File > Print Current Image in IP would print the Reference image instead
- Clearing unsaved session in DLT and choosing “Save”, would then asked whether to overwrite, despite never having saved
- Trim parameter was not stored properly in Local Measure > Thickness recipe step
- When using *Resize Image* in IP that with dynamic scale calibration, the dynamic scale will now resize to match
- Color Deconvolution no longer prevented with two classes
- Line Profile tool would become non-responsive in Local Measurements
- Multispectral images could fail to be calibrated properly
- Animate 3D Visualization > View Stereo would fail
- 3D Measure Tool would not work properly in labels view
- 3D surface rendering prevented with GPU computation enabled
- In IP Measure Tool would not work properly in some viewing modes
- Display issue in the PP when editing multiple layers at once

- Out-of-sync undo in the PP and DLT when following a multi-layer edit with a single-layer edit

Hotfixes

3.4.2.1

- Scale factor would not accept properly when entering manually in IP
- Recipe would not fully clear when individually removing all steps and measurements

3.4.2.2

- Models could fail to train in DLT

3.4.2.3

- In manual edit tools, Select > Fill/Erase now works properly.

See new features released in v3.4.0

v3.4.3 Release Notes

Bug Fixes

- Using *Replace with Triangulation* on a segmentation with one feature will no longer cause a recipe failure
- Generating a summary report out of the Post Processor will no longer produce an error
- Generating *Rotating Intercepts* measurement on an empty image will no longer produce an error
- Launching with expired maintenance will now show the Activation Window, rather than simply exiting after the notification
- Recipes that include *Adjust Contrast* developed on a workstation with a GPU will now run on a workstation without a GPU

Hotfixes

3.4.3.1

- Auto-update engine now pulls from a new source behind the scenes

See new features released in v3.4.0

v3.4.4 Release Notes

Improvements

- Added preference to normalize images before scaling to 8-bit. Preference is on by default and will continue normalizing 16-bit and 32-bit images based on their min and max when scaling to 8-bit. With preference off, 16-bit and 32-bit images will not be normalized prior to scaling to 8-bit. (See tip on Supported Formats page for added clarification.)
- Optimized performance for generation of feature measurements in the IP and PP

Bug Fixes

- All loading spinners now centered on high resolution displays

See new features released in v3.4.0

Hotfixes

3.4.4.1

- GPU Computation = Off preference now properly respected

v4.0.0 Release Notes

New Features

Curve Fitting

> Added curve fitting to the Color by Measure, Local Measurements and Histogram of Measurements windows. Users are now able to add normal, log normal and bimodal fits to their data, as well as display fit parameters. These are automatically added to the Report Generator.

Make Any Step Interruptible

> All user editable MIPAR steps can be set to interrupt recipe execution in both the image processor and batch processor. To set a step as interruptible, select the recipe step and click the ‘Lightning Bolt’ button above the recipe panel.

Session Processor

> Post Processor and Deep Learning Trainer have been combined into a single application called Session Processor. See new general layout.
>
> If you have the Deep Learning Extension, you will only see an orange app named AI Session Processor in the launch bar. Changing between DL training and measurement modes is done through the switch above the “Training” panel. See new AI layout.
>
> If you do not have the Deep Learning Extension, you will see a purple app named Session Processor in the launch bar. Layer selection editing and measurement UI has been improved. See new measurement layout..

Performance

> Support for the latest RTX 3000s NVIDIA GPUs now allows for even greater acceleration of deep learning AI training and GPU computation.
>
> Significantly improved the performance of processing large number of feature measurements, in many cases by orders of magnitude.
>
> General UX performance on Mac has been improved, especially for systems running Apple Silicon

processors.

Support Changes

- Official support for Windows 11 and macOS Monterey (12.0)
- Official support for NVIDIA GPU architecture Ampere (cc8.x) and CUDA Toolkit 11. CUDA Compute Capability 3.5 or higher now required. See GPU Computation for full support details and driver upgrade procedures.

> ! v4.0.0 uses a new version of the license manager under the hood. If after installation you face any issues with activation or software launch, please do the following:
> - Go to your license manager and look for a key marked with a red “Cloned” or “Secure Storage ID Mismatch”
> - Click the C2V button in that row
> - Email that file to support@mipar.us and let us know you need to clear clone protection after upgrading

Improvements

- MIPAR will now download a newer version of the Bio-Formats library which patches the log4j vulnerability. This library is only used by those needing to open specialty file formats. Users currently running the older version will be prompted to update on launch. We strongly recommend updating. To update manually, download the new library version, place the .jar file in [Install Directory]/plugins and restart MIPAR.
- Significantly improved the performance of processing large number of feature measurements.

Bug Fixes

- Fixed bug where the recipe directory would not update after selecting a dynamic calibration recipe.
- 2D Roundness measurement no longer improperly changes with scale factor.
- Recipes with dynamic scale factors now load properly with GPU Acceleration on.
- File>Close now properly closes the window.
- Measurement dropdown now properly populates when editing a dynamic scale calibration
- Session Processor now prints companion area column title property .
- Fixed bug in the 3D Toolbox when measuring features in 2D slices that are disjoint in that slice.
- Fixed column header for Companion Perimeter measurement in the Image Processor when a calibration is present.
- Fixed NaN output when exporting some surfaces to VTK.

- Fixed bug with Smart Cluster erroring when the execution speed was set to 1.
- Replace with Local Normal no longer errors when a feature has no pixels within the specified radius of the feature's centroid.
- Clearing unsaved session in Session Processor and choosing "Save", would then asked whether to overwrite, despite never having saved.

Detection Changes

- *Find Circles* has changed, check parameters to ensure consistent behavior between v4.0.0 and previous versions.
- Local Orientation measurements may have changed for some configurations, check for consistent behavior between v4.0.0 and previous versions.
- *Apply Deep Learning* models may have changed for some configurations, check for consistent behavior between v4.0.0 and previous versions.

Measurement Changes

- 2D Roundness measurement no longer improperly changes with scale factor
- Updated calculation of Caliper Diameter (and by extension Roundness) measurements. Previously, Caliper Diameter was measured from the center of the two pixels farthest apart in the feature. Now, Caliper Diameter considers the entire pixel area, measuring from outside pixel corners around the feature. This will increase the Caliper Diameter measurement between 1 and 1.41 pixels (for 2D measurements) or between 1 and 1.73 pixels (for 3D measurements), depending on the orientation of each feature's Caliper Diameter axis.

v4.0.1 Release Notes

Improvements

- Included new Report Generator templates with fields for curve fit outputs. You will be prompted to replace current templates on first launch of v4.0.1.0.
- *Apply Model* will no longer force a minimum size factor.
- Measure Feature tool added to Session Processor
- *Uniform Erosion* and *Uniform Dilation* can now be performed by non-integer amounts.
- Curve fitting now reports Mode.
- In the Session Processor, added arrow key shortcuts to change the current image (left/right) and active layer (up/down).

Bug Fixes

- Fixed occasional bug in Local Measurements that would prevent re-generating a measurement after changing the X-axis limits.
- Clearing a session now properly cleared the selected measurements.
- Displayed calibration properties panel now positioned properly in the Session Processor.
- When editing feature measurements from the Session Processor, changing the companion image will no longer undo changes in measurement selection.
- Removed error tone when clicking on the background of the Measurement > Select Layers window.
- Fixed bug where multiple copies of reference images would get saved in [Session Folder] > ref during portable session creation from a batch process.
- Fixed bug with displaying the reference image when crop steps are included in the recipe.
- Fixed bug with running *Grayscale Reconstruction* > Inpainting with GPU Computation on.
- Show on histogram now properly tracks with a sorted measurement table.
- Feature measurements generated in the Session Processor now respect layer selection.
- The Session Processor distance tool now displays in physical units if there is a scale factor present.
- Fixed bug with changing bin size would remove curve fit without unchecking the Fit Curve checkbox.
- In the Session Processor, removed bug when cancelling a draw operation with the Escape key.

See new features released in v4.0.0

v4.1.0 Release Notes

New Features

Load Multiple Images into Image Processor

> Image Processor can now open multiple images to combine into channels. Use File > *Open Images to Channels*.

Batch Process Multichannel Data Using Alternative Data Structure

> Batch Processor can now combine multiple images into channels. The individual channels can be accessed using the Color > *Channel Operation* recipe step.

Generate Intensity Measurements from 16-bit Images

> Image Processor can now generate pixel intensity mean, standard deviation and sum from the original bit depth image.

Improvements

- Added options to *Channel Operation* to control data normalization and divide-by-zero handling.
- Zoom is now default interactive tool in Session Processor upon opening session, and in Manual Edit in the Image Processor when adding as a new step.
- Layers will now properly align during a session merge in Session Processor as long as both sessions use the same layer names, even if they are in different orders.

Bug Fixes

- Reference Image now properly adjusted in Image Processor during Edit > Auto Crop.
- Global intensity measurements in Image Processor now properly respect their dependency on Companion Image.
- Fixed incorrect sharpen output when using GPU computation.
- Fixed bug with copy/paste of recipe steps in Image Processor.

- Fixed bug where Pre-Processing > FFT Filter step could be accepted with partial settings and would then not be editable
- Auto Segmentation memory image dependencies are now respected.
- Fixed bug where Feature Measurements set to “All” would not measure proper layers after recipe steps removed.
- Fixed bug where cancelling adding a new layer in Session Processor would incorrectly increment the default layer name.
- Fixed bug where canceling an Edit > Sparsely Sample Image step would rebuild the recipe.
- Fixed bug with running optimization of Clean-up > Reject Features.
- Fixed bug when opening pyramid Tiff with DigitalCamera and Make info fields.
- Fixed bug with cancelling a Segmentation > Manual Edit step.
- Fixed bug with cancelling a Feature Measurement step after editing.
- Fixed bug preventing the opening of images from certain camera manufacturers when pyramid or thumbnail images are present.

Hotfixes

4.1.0.1

- Added checks for starting a training with unsaved tracings and with empty layers.
- Fixed bug with “Optimize” in Image Processor when using a sub-region
- Fixed bug where original zoom limits in Image Processor would reset when changing recipe steps while zoomed

4.1.0.2

- Fixed bug where occasionally “Save Model” would not enable after a DL training in AI Session Processor.

4.1.0.3

- Memory > *Call Original* now interacts properly with Crop Image steps in Image Processor.
- Interactive measure tool in Image Processor now functions properly.
- Fixed bug where image stack would fail to create during batch process.

4.1.0.4

- Fixed bug where some steps would execute in 3D Toolbox even if preview window was cancelled.
- Fixed bug where original zoom limits in Image Processor would reset when changing viewing mode while zoomed.
- Fixed bug where AI training would occasionally error due to input image size not matching expected model size.

4.1.0.5

- Fixed bug where *Register Image* did not work properly on occasion.
- Fixed bug where *Crop Image* did not work properly on B/W images.

4.1.0.6

- Fixed bug where File > Open Recent Image would then cause File > Open Image issues.
- Fixed bug where *Load Companion Image* would fail when set as interruptible.
- Fixed bug where *Color Deconvolution* would fail on certain .CR2 images.
- Fixed open image error when loading a session with multipage TIF images.
- Fixed a bug where in the Session Processor, when an error occurs during reference replacing, the session is no longer unloadable.

4.1.0.7

- Fixed bug that caused false reports of activation failure.

v4.1.1 Release Notes

Bug Fixes

- Viewing mode stays in “Layers” after switching from Simple to Detail after loading recipe in Image Processor.
- Average neighbor distance is now properly reported for only 2 features.
- Fixed bug when loading out-of-date recipes that apply deep learning.
- Fixed bug when adding a Call Original step to a recipe while viewing Layers or Labels mode.
- Fixed bug when loading out-of-date recipes that apply deep learning.
- Fixed bug when adding a Call Original step to a recipe while viewing Layers or Labels mode.
- Fixed bug when pasting steps above a layer where the layer would no longer point to the correct step.
- Measurements no longer check for companion image size mismatch of unassigned layers.
- Resizing an image with GPU acceleration no longer turns logical images to grayscale.
- Open Images as Channels now supports logical images in Image Processor.
- Fixed bug where alternate reference images would not persist when switching recipe steps in Image Processor.
- Alternate reference images now load properly in Image Processor.

[See new features released in v4.1.0]

Hotfixes

4.1.1.1

- Fixed bug where displayed scale factor on load could be inaccurate. Actual scale factor used for measurements was unaffected.
- Fixed bug where first DCM file in folder would open regardless of which one was chosen.

Latest Release

Update:3.4.4.1
Link:MIPAR_v3.4.4.1_Update

Latest Release v4.0

Update:4.1.1.1
Link:MIPAR_v4.1.1.1_Update