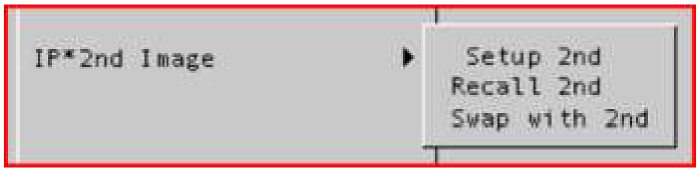
Fovea Pro 4.0 Manual

This document lists the various Fovea Pro 4.0 plug-ins with information on what they do and how to use them. More information on their various applications, and the ways to use them in typical workflow, can be found in the accompanying tutorial, the Image Processing Cookbook. In The Image Processing Tool Kit, the low-cost educational package, most of the menu selections and functions are identical but operate only on 8 bit per channel images. The plug-ins that are not included in the Tool Kit are identified.

The Fovea Pro plug-in selections are listed in the Photoshop Filters menu, and in corresponding places in other programs that accept Photoshop compatible programs. The menu entries begin with IP (Image Processing) and each has a submenu of related functions. They are dimmed out if the image is not in one of the modes that they accept, which are generally 8 and 16 bit per channel greyscale, RGB Color or Lab Color. In most cases the programs can be applied either to the entire image, or to a selection created within it using the various selection tools. Exceptions and limitations are noted for individual routines as appropriate. The various routines that display previews of processing operations will display the contents of a bounding rectangle around an irregular selection, but the final results will affect only the selected pixels.



Several of the functions in Fovea Pro require two images. This includes the Math routines (add, subtract, etc.), as well as some of the Fourier functions (deconvolution, cross-correlation, etc.) and various others (e.g., rendering an image as a surface). In all of these cases, the “second image” is not itself modified. It’s values are read, and used to modify the selected frontmost image according to the nature of the selected function. The second image is designated beforehand by making it the frontmost image and selecting the IP•2nd Image->Setup plug-in. Physically, this copies the current image into a private memory storage, and written to disk as a file. It is then not even necessary for the image to remain open when the subsequent two-image operation is performed. The second image is remembered, unchanged, until a different image is designated as the second image.

-> Setup 2nd

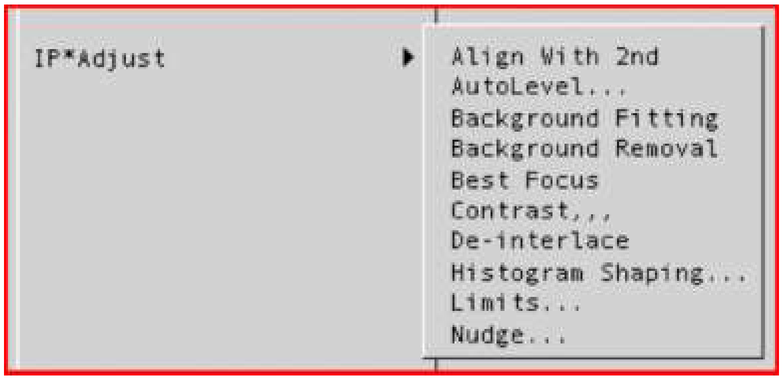
This selects the current active (or frontmost) image as the second image, and writes its contents to a private disk file. Note that if an image in one mode (e.g., 16 bit grey scale) is selected as the second image, and some later use of it requires some other image mode (e.g., 8 bit RGB color), the data conversion is performed automatically. Also, if the dimensions of the designated second image are not the same as those of the image it is later used with (e.g., for subtraction), the common rectangular area at the upper left corner of the two images is used. Finally, if an irregular arbitrary selection has been made on the frontmost image when this function is selected, the minimum bounding rectangle around the selection region is actually saved. When operations such as addition are performed, the region marked on the current active (frontmost) image is used.

-> Recall 2nd

This reads the stored second image file and overwrites the current image with its contents (or, if they are not the same size, overwrites the common rectangular area in the upper left corner). One use of this function is to convert image modes, since if the current front image has a different mode than the original stored image, the data values are converted automatically.

-> Swap with 2nd

This reads back the stored second image and simultaneously writes the current front image into the second image memory. It is primarily used in creating actions that carry out complex sequences of processing operations.



The adjustment routines comprise a variety of functions used to correct various defects that typically arise in image capture.

-> Align with 2nd

If two images are to be combined, for instance to subtract one from another, it is necessary that they be aligned. Minor shifts in camera position, or microscope stage drift, can result in image offset. This routine uses cross-correlation of the image with the stored second image to detect the amount of that offset (presuming that they are images of similar structures). The front image will be displaced in X and/or Y position to the location that best corresponds to the position of the previously stored second image, to sub-pixel accuracy. This routine does not correct for changes in rotation or scale (see Register to 2nd).

-> AutoLevel

Images with nonuniform illumination (e.g., due to off-axis lighting, optical vignetting, etc.) can be “leveled” in several different ways. The Bright and Dark options are used when the brightest (or darkest) features in the image are well distributed across the image area and should all have the same brightness. The function finds those features, fits a polynomial function to their brightness values, and uses that function to level the overall image brightness. Finding both bright and dark features and fitting both polynomials allows linearly stretching the brightness values between those limits everywhere in the image. This adjusts the pixel values differently at each point so that brightness and contrast are made uniform across the image.

-> Background Fitting

The automatic methods in AutoLevel Bright and Dark assume that either the brightest or darkest features present in the image should be made uniform, and are uniformly dispersed across the image area. When that is not the case, it is often possible for the user to manually select some representative regions on the image (preferably well distributed spatially) that should be uniform. This is commonly done either with the manual outlining tools (lasso, polygon) or with the wand, using the shift key to add additional regions to the selection. When the selections have been made, this function performs the polynomial fit and saves the data for use in the Background Removal function.

-> Background Removal

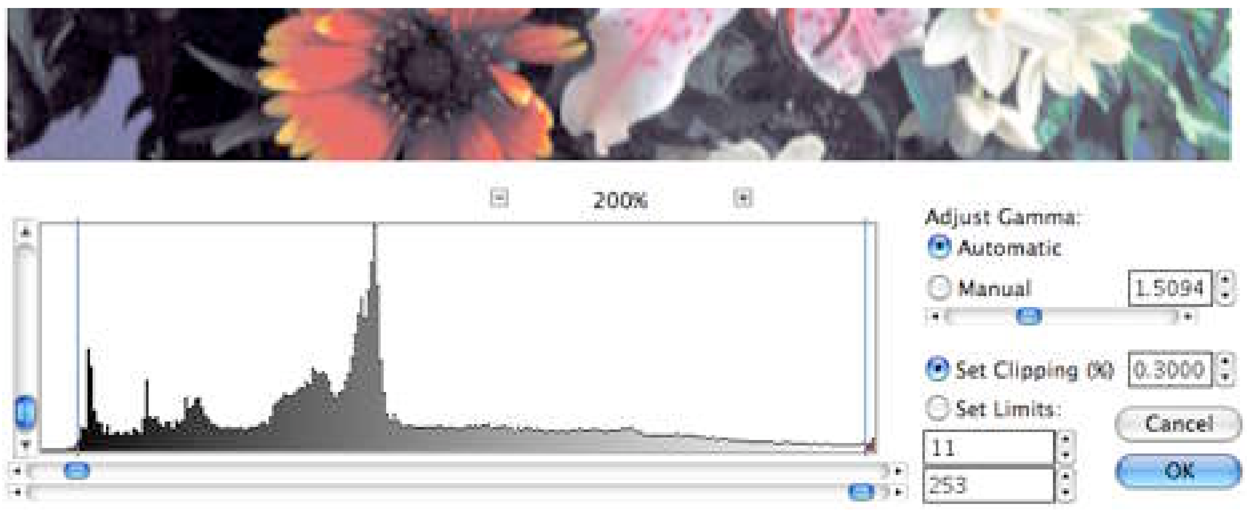
Once a background polynomial has been established using the Background Fitting function, the shading can be removed and the image brightness leveled using this function. The selection used to define the background regions should be removed (Select->None or Command/Control-D) so that the removal is applied to the entire image area. The function is saved and can be re-applied to additional images.

-> Best Focus

If two (or more) images are taken with different focus settings, it is possible to combine them to construct an “extended focus” image. The procedure selects the pixels from two images (the current front image and the stored second image) keeping whichever is judged to have the better focus based on the local variance, and selected pixels from the second image are inserted into the front image. The function can be repeated with a series of images to construct the final result. The images must be aligned beforehand.

-> Contrast

This function is very much like the Photoshop Image->Adjustments->Levels function except that the Auto button also adjusts the Gamma value on grey scale images for optimum contrast (so that the midpoint of the histogram is set to medium grey). Also, for color images, it operates on the intensity data while leaving the color (hue and saturation) unchanged, whereas the built-in function operates separately on the RGB channels and can therefore result in color alterations.



Adjust Gamma: [✓ Automatic] Manual 1.5094

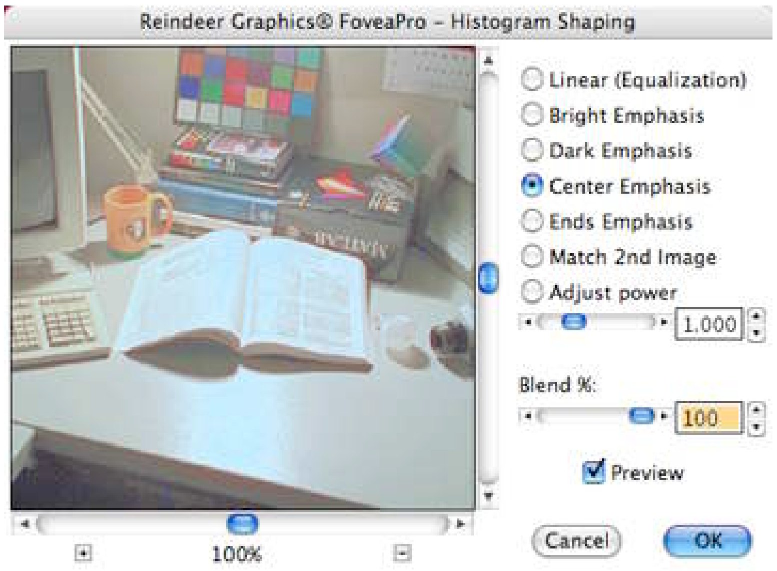
[✓ Set Clipping (%)] 0.3000

Set Limits: 11 253

-> DeInterlace

Standard video cameras scan two "fields" to build up one "frame," by alternately scanning the even and then the odd lines of the image. Either because of camera or scene motion or electronic timing jitter, these two fields can be slightly offset. This function corrects that by using the same logic as the "Align to 2nd" function to find the best alignment of the two fields and apply the shift so that the even and odd scan lines are best aligned with each other.

-> Histogram Shaping



Reindeer Graphics® FoveaPro – Histogram Shaping

- Linear (Equalization)
- Bright Emphasis
- Dark Emphasis
- [✓ Center Emphasis]
- Ends Emphasis
- Match 2nd Image
- Adjust power

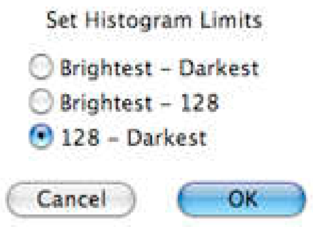
1.000

Blend %: 100

[✓ Preview]

This function includes several options for adjusting the image contrast in a way that depends on the actual image content. Linear shaping (classic histogram equalization) is similar to the Photoshop Image->Adjustments->Histogram Equalization routine. The other options offer additional choices besides a strictly linear cumulative result. Bright and Dark emphasis produce logarithmic or exponential overall contrast that is particularly useful with images acquired using linear devices, to make the results more film-like. Middle or Ends Emphasis are suitable for images with low or high inherent contrast (they correspond to Gaussian and error function curve shapes). The adjustable power setting corresponds to the exponent of a power law shape for the cumulative histogram, ranging from 0.25 (dark emphasis) to 3.0 (bright emphasis). Match 2nd Image adjusts the pixel brightness values so that the histogram matches that of a stored 2nd image.

-> Limits

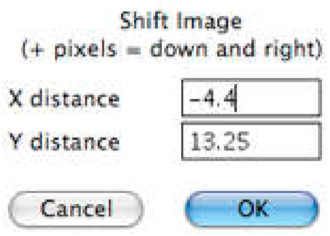


Set Histogram Limits

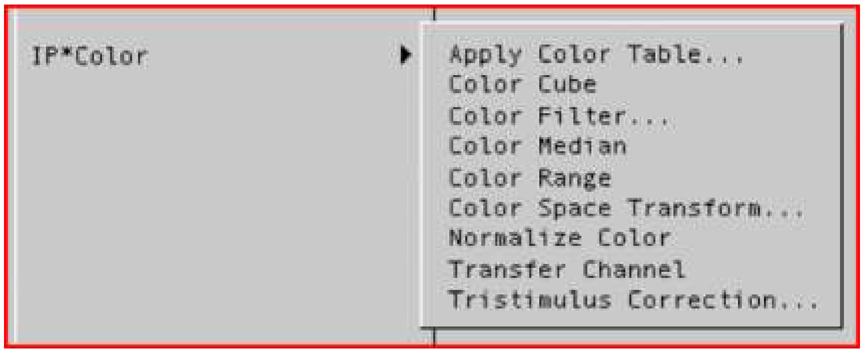
- Brightest – Darkest
- Brightest – 128
- [✓ 128 – Darkest]

This function is primarily useful after the Rank->Top Hat filter, or in some instances with Fourier power spectra. It offers a one-click way to set the brightness limits in the image to the exact end points of the histogram, or to medium grey. For example, applied after the Top at filter, this routine can select just the bright or dark features that were selected by the top hat. The same operations can be carried out using the Contrast plugin but this plug-in is more efficient, particularly for use in automated action sequences.

-> Nudge


Shift Image
(+ pixels = down and right)
X distance -4.4
Y distance 13.25

By placing images into Photoshop's Layers and adjusting the top layer opacity, it is possible to manually align one image to another. However, clicking and dragging, or using the keyboard arrow keys, can only move the image in increments of whole pixels. This function allows for sub-pixel alignment in X and Y.



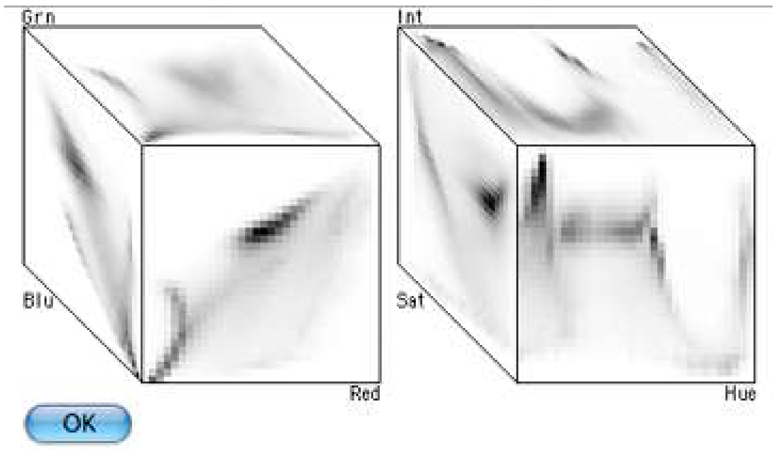
Photoshop supports RGB, CMYK, and Lab color spaces. For many image processing purposes, a HSI (Hue-Saturation-Intensity) color space offers advantages. This menu collects together a number of functions that supplement the built-in routines, and provide specific image manipulations in HSI space. In general these functions are inappropriate for grey scale images and cannot be applied to them.

-> Apply Color Table

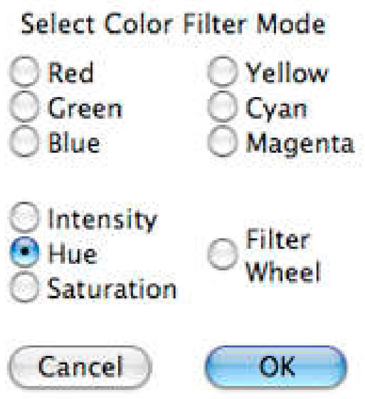
Photoshop allows an 8 bit grey scale image to be converted to Indexed color mode, and a color table (CLUT) to be applied so that each of the 256 grey scale values is shown as an arbitrary color. This function uses those same CLUT files (file type *.act) to be applied in a different way. If the grey scale image (either 8 or 16 bit) is first converted to RGB color, and then this function is selected, the CLUT file will be loaded and used to convert the brightness values to color. The colors are smoothly interpolated for 16 bit images.

-> Color Cube

This function displays the RGB and HSI components of an image in three-dimensional histogram form, as an informational guide. Images in which the clusters of color values are well dispersed are usually easier to segment.



-> Color Filter



The built-in Photoshop color channels dialog allows selecting the red, green or blue channel. This plugin also allows selection of the hue, saturation or intensity channel from a color image. In addition, an arbitrary color filter can be selected which is then combined with each pixel in the image (technically, a vector dot-product is performed between the pixel color and the filter color). This produces the same effect as placing a filter of that color in front of the camera. Reducing the image to grey-scale afterwards produces the same effect as filtering in front of a black-and-white camera.

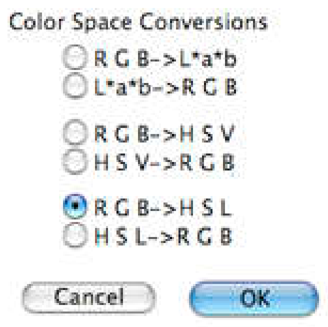
-> Color Median

The usual median filter, both in Photoshop and in the IP•Rank menu, ranks the pixel values according to brightness in order to select the median. For color images, it is also possible to perform a true color median. That selects the median value as the pixel in the neighborhood whose color coordinates lie closest in a vector sense to all of the others (i.e., the sum of the vector distances from its coordinates to all the others is minimum). This filter implements that method.

-> Color Range

The usual range filter (under the IP•Rank menu) calculates the brightness difference between the lightest and darkest pixels in the neighborhood, without regard to their color values. This plug-in instead calculates the difference between pixels in a vector sense in color space to find the maximum difference. It is primarily used to find edges or boundaries in images, which may involve differences in color as well as brightness of regions.

-> Color Space Transform



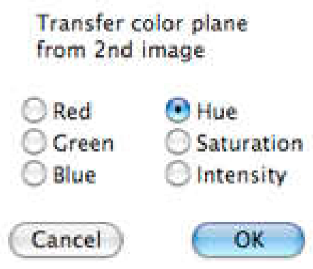
This plug-in converts between the RGB, Lab, HSL (hue-saturation-luminance) and HSV (hue-saturation-value) color spaces. It is typically used before selecting the Photoshop Split Channels function to separate the color image into three grey scale images. Because Photoshop and the computer's display hardware do not know how to display these images with other sets of color coordinates, they still interpret the three channels as though they were red, green and blue. This results in a display that is visually incorrect and distracting, but after separation the three channels can be correctly interpreted and processed. The names assigned by Photoshop to the split channels will be red, green and blue, but the actual values they contain will be those from the selected color space. After manipulation of these images, it is possible to recombine them and convert them back from their current color space to RGB for correct display.

-> Normalize Color

This function is closely related to the Tristimulus correction listed below. It attempts to find the reddest, bluest and greenest patches of color in the image (expecting a standard color chart), measures the amount of cross-channel intensities (e.g., how much green intensity there is in the blue channel), and calculates and applies the tristimulus matrix in order to make the colors throughout the image correct by adjusting the red, green and blue values in the color chart patches. In a complex image, it may be necessary to place a selection around the color chart patches so that the plug-in finds the correct colors. Then Undo the result, select the entire image, and use the Tristimulus function listed below to apply the same correction matrix (which has been remembered by the software) to the entire image.

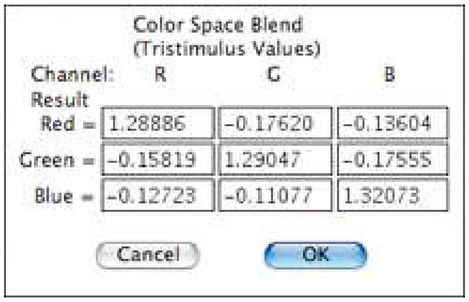
-> Transfer Channel

This routine makes it easy to assemble color images or to manipulate individual color channels. Place a grey scale image in the second image memory (IP•2nd Image->Setup 2nd Image) first. Then use this function to insert those grey scale values into the selected channel of the front image. When this is done to the red, green or blue channel the result is the same as could have been produced in Photoshop by splitting the channels, replacing one by the new data, and then merging them back together (although this function is much quicker and easier). But the plug-in also makes it possible to directly insert the intensity information into the hue, saturation or intensity channel of a color image, with the conversion to and from RGB format occurring automatically.


Transfer color plane
from 2nd image

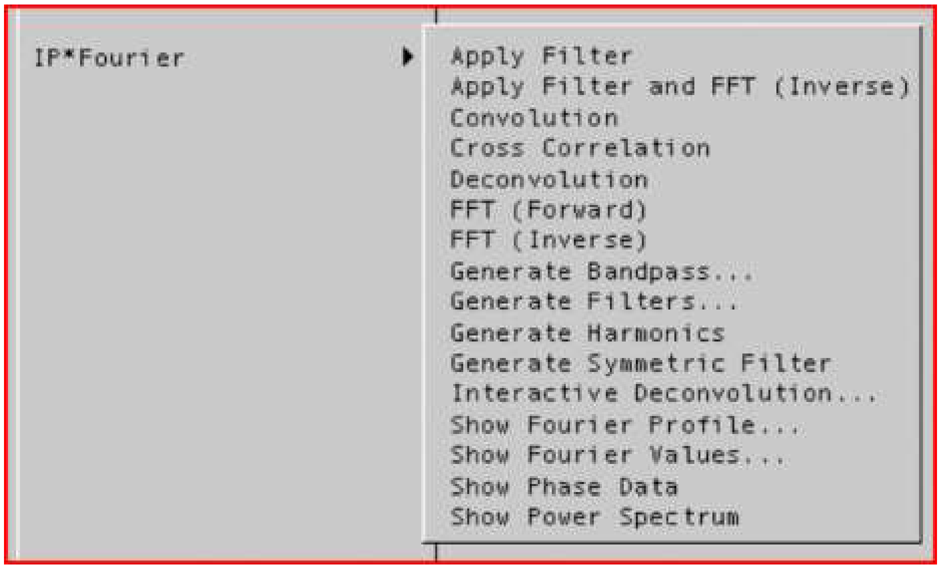
Red [✓ Hue]
Green Saturation
Blue Intensity

-> Tristimulus Correction


Color Space Blend
(Tristimulus Values)

Channel: Result	R	G	B
Red =	1.28886	-0.17620	-0.13604
Green =	-0.15819	1.29047	-0.17555
Blue =	-0.12723	-0.11077	1.32073

This routine applies a correction matrix to the red, green and blue components of each pixel. The matrix may be determined from a color chart with standard RGB patches (see Normalize Color, above), or any other set of three known colors by reading the values of red, green and blue present in each one and constructing a matrix showing the cross-channel intensities. The inverse of that matrix is the tristimulus correction matrix, which is remembered from the last use of Normalize Color, or which can be calculated (e.g., with Excel) and entered by hand.



The Fourier Transform routines all use a high-precision floating point file the contains the real and imaginary values that constitute the transform. There is only one such file, and only one such transform, which is created by performing a Forward FFT (and replaced by performing another one). The display of the power spectrum is derived from the transform file, but manually editing it does not alter the stored values. The various filters (or masks) that are created by some of the routines are normal 8 or 16 bit images in which the grey scale represents a real value from 0 (white) to 1 (black). They can be applied to the stored transform. Many of the routines require that the image have dimensions that are the same as the stored transform, and are an exact power of two (e.g., 64, 128, 256, 512, 1024...). Exceptions are noted.

-> Apply Filter

This routine applies the pixel brightness values in the current front image to the amplitude of the stored Fourier transform, and displays the resulting power spectrum. Black is interpreted as 1 and white as 0, with proportional grey scale values. The image must have the same dimensions as the transform. This is typically used to apply filters that are created either manually or using the Generate... routines under this menu.

-> Apply Filter and FFT (Inverse)

This routine applies the front image as a filter to the stored transform and also performs an inverse Fourier transform to generate the real-space (or pixel-space) image. This is equivalent to using Apply Filter and FFT (Inverse) as separate functions, but is faster and also has the advantage that the stored transform file is not modified in the process (allowing other filters to be tried with the original transform data).

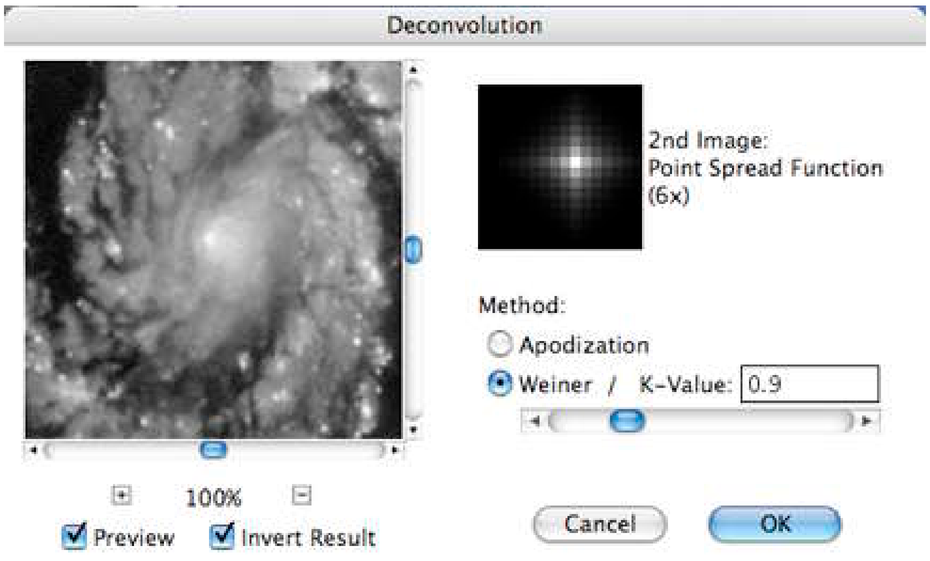
-> Convolution

Convolution in Fourier space is equivalent to convolution performed in the pixel domain (e.g., by using a kernel of values in the Custom filter). For large kernels, the Fourier-space method is faster. The routine requires two images, one of which is the current front image and the other is the current second image (which must be set up before using this function). The two images do not have to be the same dimension, and the dimensions are not restricted to powers of two. The result is a real-space (pixel) image.

-> Cross-Correlation

Cross-correlation in Fourier space is similar to convolution except for a reversal of the phase values of the complex values. It is typically used to locate features in an image that have a similar pattern of pixel brightnesses as those in a second image. The routine requires two images, one of which is the current front image and the other is the current second image (which must be set up before using this function). The two images do not have to be the same dimension, and the dimensions are not restricted to powers of two. The result is a real-space (pixel) image.

-> Deconvolution



Method: [Apodization] [✓ Weiner] / K-Value: 0.9

[✓ Preview] [✓ Invert Result]

Just as convolution is a multiplication of the complex values in the Fourier transform of the image by those in the Fourier transform of the second image, so deconvolution is a division. This is typically used to remove the effects of a convolution, most often one produced by the imaging system (such as motion or out-of-focus blur). If an image of the “point spread function” due to the imaging system is available, deconvolution will remove the blur from the image and restore sharpness. The amount of blur removal is limited by the noise in the image. There are two ways to deal with this. One (apodization) omits those parts of the Fourier transform in which the division would produce a numerical overflow. The second (the Wiener constant) adds a constant to the divisor to prevent overflow. This value can be set interactively in the dialog. Too small a constant will cause noise to dominate the result, while too large a value will limit the amount of sharpening that can be obtained. The routine requires two images, one of which is the current front image and the other – the point spread function – is the current second image (which must be set up before using this function). The two images do not have to be the same dimension, and the dimensions are not restricted to powers of two. The result is a real-space (pixel) image.

-> FFT (Forward)

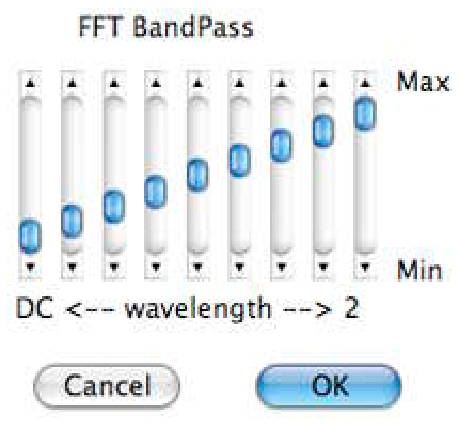
This routine calculates the Fourier transform of the current image and places the real and imaginary values into the stored transform array (overwriting any previous transform). The image must have dimensions that are an exact power of two. Images with other dimensions can be enlarged to the next greater exact dimensions using the Photoshop Image->Canvas Size selection, or a selection marquee of the correct size can be placed within the image. The power spectrum of the transform is displayed, with darker values representing larger amplitudes. The DC term is at the center of the power spectrum, and frequency increases with radius.

-> FFT (Inverse)

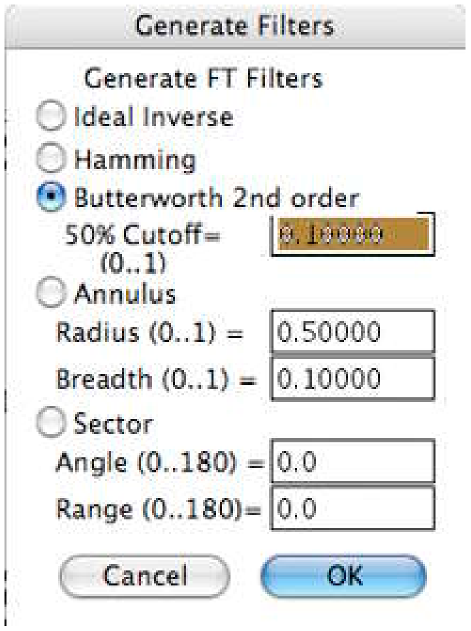
This routine calculates the inverse Fourier transform from the stored transform array and places the resulting real-space image into the front image. The image must have the same dimensions as the stored transform data. The stored transform is not modified by this procedure.

-> Generate Bandpass

Filters (or masks - the terms are used here interchangeably) can be multiplied by the amplitude of values in the stored Fourier transform array by the Apply Filter function above. The filter is an image (with the same dimensions as the stored transform array) in which white is interpreted as zero (hence zeroing the amplitude of the corresponding frequency term in the Fourier transform), black is interpreted as one, and grey values provide proportional control. This function offers one way to generate a filter. The sliders correspond to different frequencies, much like the "equalizer" sliders on an audio amplifier. They can be used to generate a filter that will selectively keep or erase frequencies from the transform.



-> Generate Filters


Generate Filters

Generate FT Filters

Ideal Inverse

Hamming

[✓ Butterworth 2nd order]

50% Cutoff= (0..1) 0.10000

Annulus

Radius (0..1) = 0.50000

Breadth (0..1) = 0.10000

Sector

Angle (0..180) = 0.0

Range (0..180)= 0.0

This function generates several different kinds of filters, including a second order Butterworth filter with a user-specified crossover frequency (expressed as a fraction of the maximum radius), Hamming and ideal inverse frequency filters, an annular filter (with cutoffs expressed as a fraction of the maximum radius), or a sector filter (with cutoffs specified as angles). In some cases, several different filters can be created and combined with a Boolean AND or OR (e.g., to select a range of angles and frequencies), and then slightly blurred (with a Gaussian blur) to minimize Gibbs ringing.

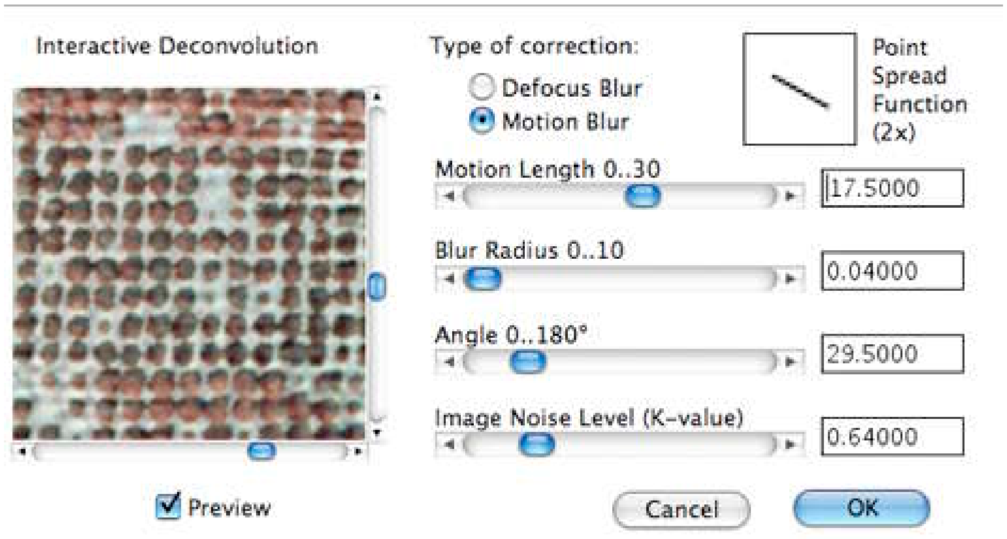
-> Generate Harmonics

Fourier transforms of images containing periodic structures or noise typically produce a power spectrum with a series of regular spikes. There are several ways to process the power spectrum image (e.g., a top hat filter) to locate these spikes and produce a filter that will keep or eliminate the periodic information. However, another method that is often used it to mark the one or two lowest frequency spikes using the pencil tool (with a suitably large diameter and a black color). Then this routine will locate those marks and generate the full set of harmonics. These appear as small black dots, which can be thresholded, optionally dilated to cover the spikes, optionally smoothed with a Gaussian blur to reduce Gibbs ringing, and then used as a filter.

-> Generate Symmetric Filter

The convention used in plotting the power spectrum duplicates the information in the top half with rotational symmetry about the DC point in the center. If manual marking of points to be kept or removed is performed using a black pen (so that thresholding will produce a suitable filter), it is only necessary to mark locations in any half of the display. This routine will locate the black points marked and duplicate them on the other half of the display.

-> Interactive Deconvolution

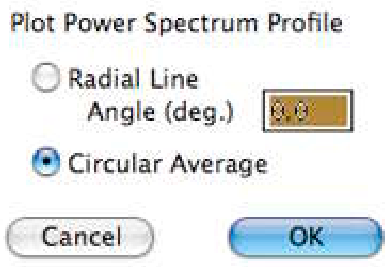


Motion Length 0..30: 17.5000
Blur Radius 0..10: 0.04000
Angle 0..180°: 29.5000
Image Noise Level (K-value): 0.64000

In cases where an image of the point spread function is not available for deconvolution, it is often practical to model it. Defocus blur is typically modeled by a Gaussian shape, optionally with some astigmatism (elongation of the Gaussian by an arbitrary amount and direction). Motion blur is typically modeled by a straight line, optionally with some lateral blur. This plugin allows the user to interactively generate a model point spread function and observe the result of deconvolving the image with it. The Wiener K-factor is the same as that described above for the Deconvolution routine. The FFT of the front image is calculated internally and does not overwrite the stored Fourier transform array, and there is no restriction of the image dimension to an exact power of two. Interactive deconvolution is not included in the Image Processing Tool Kit.

-> Show Fourier Profile

This routine plots the average value of the amplitude of the Fourier transform data as a function of radius (frequency). The plot is displayed and the numerical values are written to a tab-delimited ascii text file on disk which can be opened in a spreadsheet or other data analysis program.

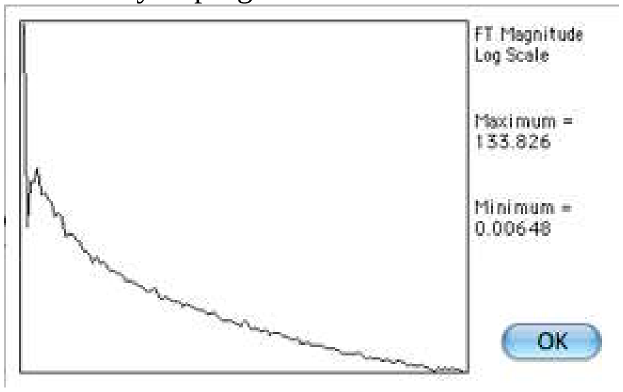


Plot Power Spectrum Profile

Radial Line
Angle (deg.) 0.0

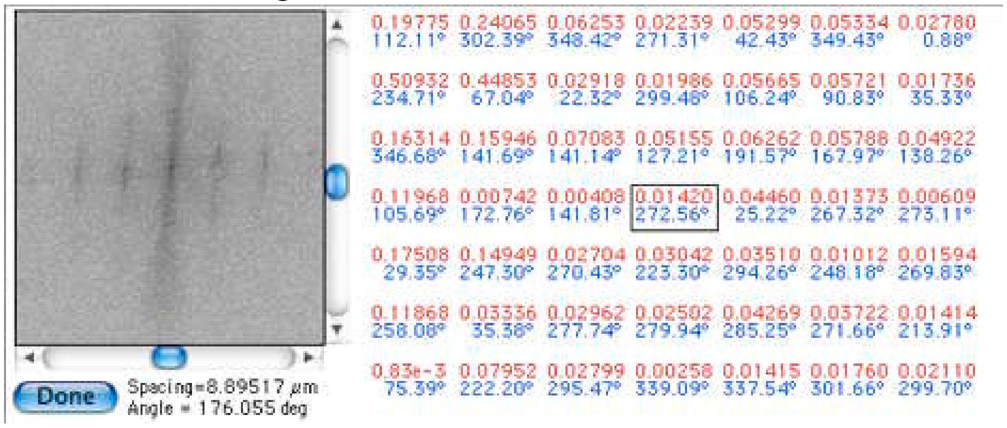
[✓ Circular Average]

!



Show Fourier Values

This function is used to access the stored Fourier transform values. Clicking on any point in the power spectrum display in the preview will show the complex values in the transform (expressed as amplitude and phase). The coordinates of the point clicked are converted to an angle and spacing of the corresponding structure which provides an efficient method to measure structural dimensions from the Fourier transform. This conversion makes use of the image calibration previously established using the IP•Measure Global->Calibrate Magnification routine.



0.19775	0.24065	0.06253	0.02239	0.05299	0.05334	0.02780
112.11°	302.39°	348.42°	271.31°	42.43°	349.43°	0.88°
0.50932	0.44853	0.02918	0.01986	0.05665	0.05721	0.01736
234.71°	67.04°	22.32°	299.48°	106.24°	90.83°	35.33°
0.16314	0.15946	0.07083	0.05155	0.06262	0.05788	0.04922
346.68°	141.69°	141.14°	127.21°	191.57°	167.97°	138.26°
0.11968	0.00742	0.00408	0.01420	0.04460	0.01373	0.00609
105.69°	172.76°	141.81°	272.56°	25.22°	267.32°	273.11°
0.17508	0.14949	0.02704	0.03042	0.03510	0.01012	0.01594
29.35°	247.30°	270.43°	223.30°	294.26°	248.18°	269.83°
0.11868	0.03336	0.02962	0.02502	0.04269	0.03722	0.01414
258.08°	35.38°	277.74°	279.94°	285.25°	271.66°	213.91°
0.83e-3	0.07952	0.02799	0.00258	0.01415	0.01760	0.02110
75.39°	222.20°	295.47°	339.09°	337.54°	301.66°	299.70°

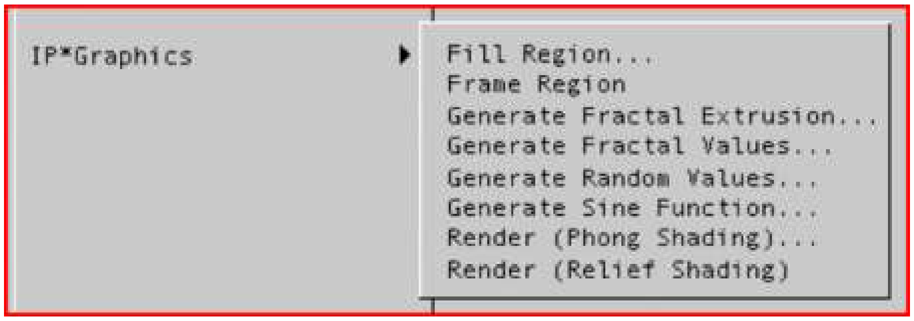
Spacing=8.89517 µm
Angle = 176.055 deg

-> Show Phase Data

This function displays the phase values from the stored Fourier transform, using the grey scale values 0..255 to represent phase angles from 0 to 2" . It requires that the image have the same dimensions as the stored array. It is sometimes useful to assign spectrum color values (see IP•Color->Apply Color Table) to the resulting display to assist in visualizing the phase angles.

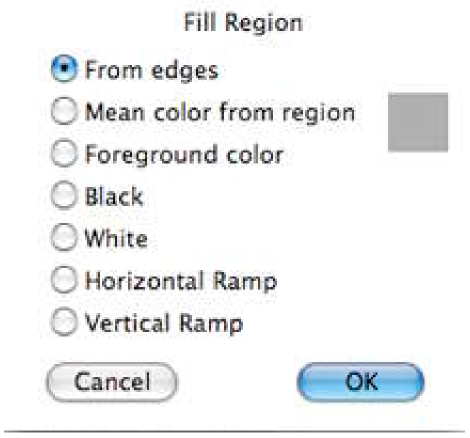
-> Show Power Spectrum

This function displays the power spectrum from the stored Fourier transform. It requires that the image have the same dimensions as the stored array. Typically, this routine is used after the user has tried creating several different filters and wishes to see the power spectrum again as a guide to modifying them. The same power spectrum display is generated by the original Forward FFT function.


IP*Graphics
Fill Region...
Frame Region
Generate Fractal Extrusion...
Generate Fractal Values...
Generate Random Values...
Generate Sine Function...
Render (Phong Shading)...
Render (Relief Shading)

-> Fill Region

This function will fill the interior of a selection, or the entire image if no selection has been made. The choices include the average value of the pixels originally there, black, white, and vertical or horizontal ramps of grey scale values. It is also possible to fill by smooth interpolation from the edges (analogous to fitting an elastic membrane to a hoop). It is important to remember that the filling is done using the edge pixels within the region, so in some cases it will be necessary after defining a selection to expand it by one pixel (Select->Modify->Expand) so that the border is included. Regions used for this purpose (and most others) should not have antialiased edges.

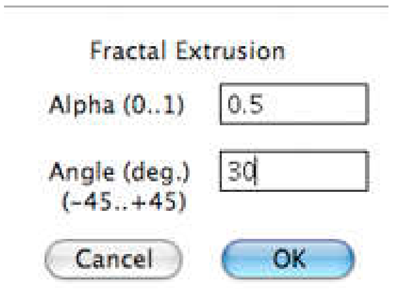

Fill Region

[✓ From edges]
Mean color from region
Foreground color
Black
White
Horizontal Ramp
Vertical Ramp

-> Frame Region

This function colors the edge pixels in a selection region (or the entire region if no selection has been made) in the current pencolor. It is similar to the Photoshop Edit->Stroke function except that the latter routine anti-aliases the line by blending the pen color with the current pixel contents, and the plugin does not. This is important because a solid outline is needed for various measurement or thresholding purposes.

-> Generate Fractal Extrusion


Fractal Extrusion

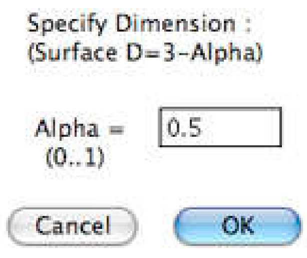
Alpha (0..1) 0.5

Angle (deg.) (-45..+45) 30

This routine generates a fractal profile with a user-specified dimension, and then sweeps it across the image in a chosen direction. It provides a way to model various types of linear structures or defects, including scratches.

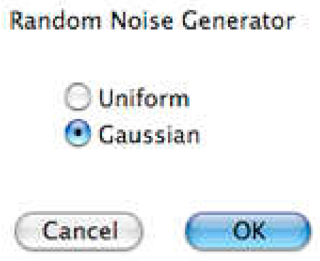
-> Generate Fractal Values

This routine generates pixel grey values that have a fractal distribution, with a user-specified dimension. Unlike random pixel values, the fractal ones have a spatial correlation (pink noise). Rendering a surface from such an image produces a realistic topography.


Specify Dimension :
(Surface D=3-Alpha)

Alpha = (0..1): 0.5

-> Generate Random Values

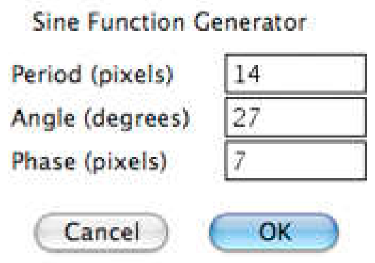

Random Noise Generator

Uniform [✓ Gaussian]

This function generates random pixel values (either Gaussian or rectangular random functions). This is a model for white random noise. Typically noise images produced by this and the other generators are blended with other images

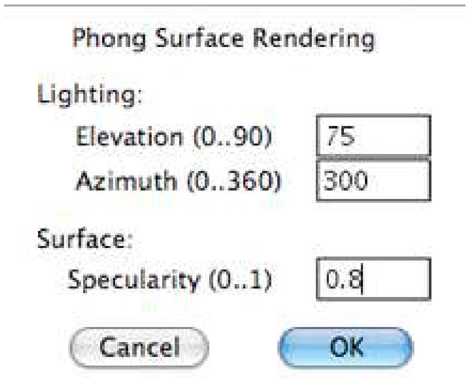
-> Generate Sine Function

This function generates periodic, sinusoidal lines with a user-specified spacing and orientation. This is a model for periodic noise. All of the four “Generate...” plugins are typically used to generate values that can then be blended singly or in combination with images.


Sine Function Generator

Period (pixels)	14
Angle (degrees)	27
Phase (pixels)	7

-> Render (Phong Shading)


Phong Surface Rendering

Lighting:

Elevation (0..90)	75
Azimuth (0..360)	300

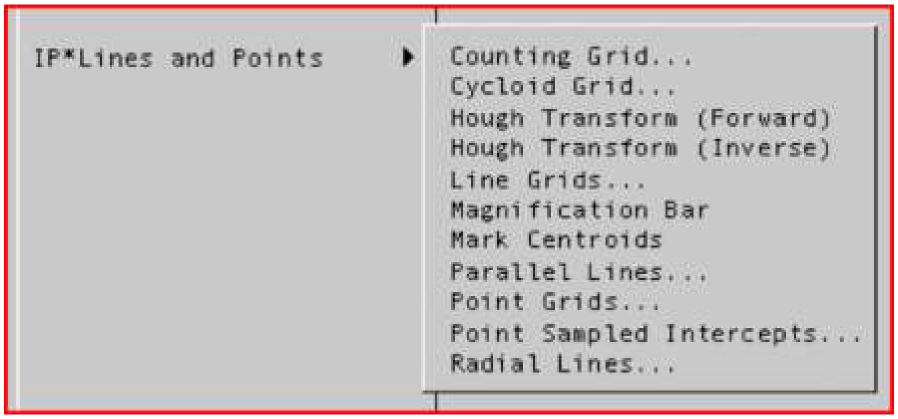
Surface:

Specularity (0..1)	0.8

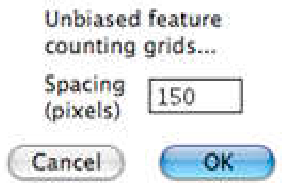
The brightness values in the image are interpreted as representing elevation, and the resulting surface is rendered using Phong shading, with user control over the placement of the light source (elevation above the horizon and compass direction) and the specularity of reflections from the surface (1.0 corresponds to a perfect metal, 0.0 to a perfectly diffuse scatterer, 0.5 to something like plaster).

-> Render (Relief Shading)

This function is similar to the Photoshop “embossing” function, producing bright and dark shadows adjacent to steps in brightness so that the image has a three-dimensional appearance. If the image is first changed to RGB Color mode, the embossed relief appears in the intensity channel while the original brightness values are represented by hue. This is particularly useful for range images in which brightness represents elevation.



-> Counting Grid

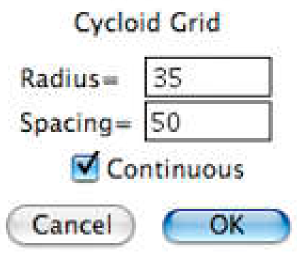


Spacing (pixels) 150

This routine draws a stereological counting grid (with exclusion lines) on the image using the current pen color and reports the area in the current calibration units. The correct unbiased procedure for counting number per unit area is to include features that lie partially or entirely within the marked region but which do not touch the exclusion line. Typically this grid would be used with manual marking by selecting a unique pen color and using the pencil tool with a diameter that makes visible but not overlapping marks to place a mark on or next to each feature to be counted. Then select IP•Measure Features->Count to count the marks.

-> Cycloid Grid

This routine draws cycloids on the image using the current pen color and reports the total line length. The orientation of the cycloids corresponds to rolling the generating circle along a horizontal line, which is correct for the usual image orientation with the vertical direction of sectioning vertical on the screen. The user can select the diameter of the generating circle and the spacing between lines, and whether to draw continuous lines or disconnected arcs. The usual criterion is that the lines should be sparse enough to not sample the same stereological feature multiple times. These lines are typically used either for manual marking and counting, or for automatic counting by combining the lines with feature outlines using a Boolean AND and counting the intersections.



Radius= 35
Spacing= 50
✓ Continuous

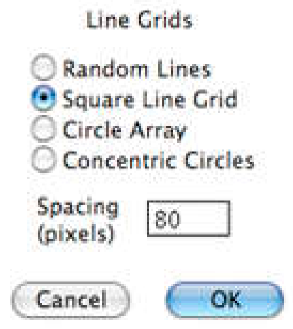
-> Hough Transform (Forward)

The linear Hough transform is a tool for finding alignments of features or points. This function works best with a small number of points, so after thresholding, the ->Mark Centroids routine may be useful to mark each feature with a single point. Before using this routine, place the image into the second image memory (IP•2nd Image->Setup) and create a new image of whatever size is needed to provide the desired precision. Then select the function, which will draw a sinusoid in the Hough space image for each point in the original. These sinusoids add together, so the crossing points can be detected either by image processing or manually.

-> Hough Transform (Inverse)

This routine reconstructs the lines in real- or pixel-space that correspond to points in Hough space. After the crossing points in the Hough space image have been thresholded or marked so that they are black (value = 0), place the Hough space image into the second image buffer. Then either select the original pixel image (or a duplicate of it) and run the routine. For each exactly black point in the Hough space image, the corresponding line will be drawn onto the real space image, showing the linear arrangement of features.

-> Line Grids

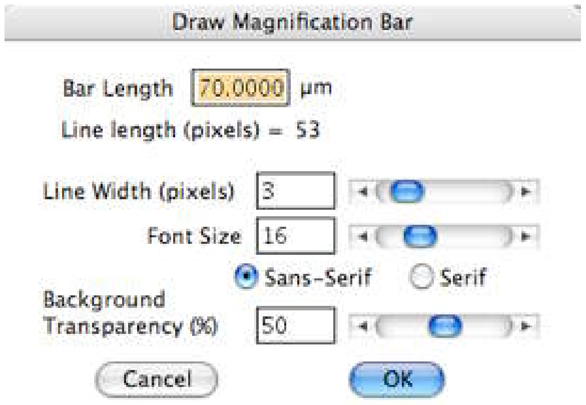

Line Grids

Random Lines [✓ Square Line Grid] Circle Array Concentric Circles

Spacing (pixels): 80

This routine draws various line grids onto the image using the current pen color, and reports their total length in the current calibration scale. The user can select a square grid, concentric circles or a grid of circles, or random lines. The "spacing" value controls the density of lines (except for the random case). These lines are typically used either for manual marking and counting, or for automatic counting by combining the lines with feature outlines using a Boolean AND and counting the intersections.

-> Magnification Bar


Draw Magnification Bar

Bar Length: 70.0000 µm

Line length (pixels) = 53

Line Width (pixels): 3

Font Size: 16

[✓ Sans-Serif] Serif

Background Transparency (%): 50

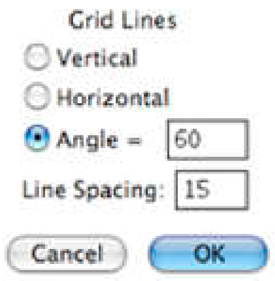
This routine draws and labels a magnification bar in the lower right corner of the image that corresponds to the current calibration units and scale. It is sometimes convenient to do this in a new blank image or layer so that the resulting marks can be copied and pasted in the location desired on the original. The default length of the bar depends on the size of the image in which it is drawn. The user can control the line length and thickness, the size of the lettering and the degree of transparency. The current pencolor is used.

-> Mark Centroids

This routine can be applied to a thresholded binary image to mark the centroid location of each individual feature present (a feature is defined as a touching set of non-white pixels). Since the features are erased in the process, it is often convenient to apply this function to a duplicate image or duplicate layer. The centroid locations can be used for a variety of measurement or processing procedures. Note that the centroid of a non-convex feature may not lie within the feature itself. The IP•EDM->Ultimate Points routine marks points that are within the feature boundaries, in cases for which that is required.

-> Parallel Lines

This routine draws sets of parallel line grids using the current pen color, and reports their total length in the current calibration scale. The user can select horizontal or vertical lines, or enter an arbitrary orientation angle, and can also specify the line spacing (in pixels). These lines are typically used for automatic measurement by combining the lines with thresholded binary features using a Boolean AND and measuring the line segments.



Grid Lines

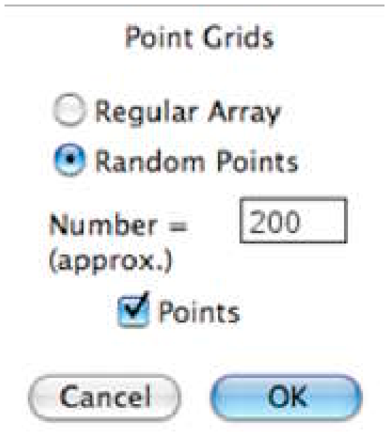
Vertical

Horizontal

[✓ Angle =] 60

Line Spacing: 15

-> Point Grids



Point Grids

Regular Array

[✓ Random Points]

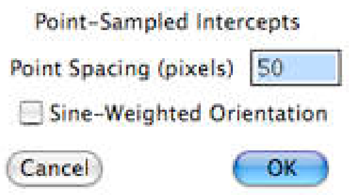
Number = (approx.) 200

[✓ Points]

This routine places a grid of points onto the image using the current pen color, and reports the exact number of points. The user can select random or regular grid arrays and the requested number of points. For the regular arrays, the actual number may be less than the requested number as required by the shape and dimension of the image. These points are typically used for automatic stereological measurement by combining the lines with thresholded binary features using a Boolean AND and counting the number of hits. For manual marking and counting, it is usually preferable to have the routine mark small circles instead of points, so that the image information at the exact point location is visible.

-> Point Sampled Intercepts

This function is used for the stereological point-sampled intercept measurement procedure. It generates a regular grid of points with the user-specified spacing, and for each point that falls on a feature (any non-white pixel), it draws a line to the edge of the feature. These lines are drawn at angles that are either uniformly distributed or have sine-weighted distributions (appropriate for vertical section images) as selected by the user. The length of each line from the originating grid point to the feature edge is determined, and the mean value of the cube of this "radius" is reported in the current calibration units. For procedures such as determining the variance of a feature size distribution, this value should be multiplied by (4" /3) to obtain the volume-weighted mean volume.

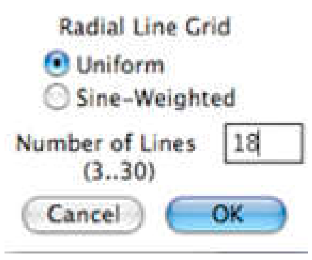


Point-Sampled Intercepts

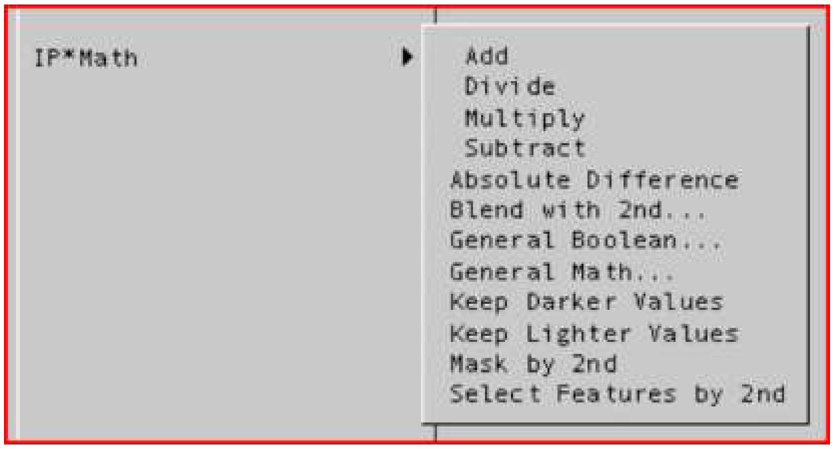
Point Spacing (pixels) 50

Sine-Weighted Orientation

-> Radial Lines



This routine draws sets of radial line grids using the current pen color, and reports their total length in the current calibration scale. The user can select the number of lines to draw and whether they should be spaced uniformly in angle, or sine weighted. The latter case applies to vertical sections in which the vertical sectioning direction is vertical on the screen, and produces lines for which the sine of the angle increases by a constant increment. These lines are typically used for automatic stereological measurement by combining the lines with the outlines of thresholded binary features using a Boolean AND and counting the intersections. The uniform lines may be used for these applications with random sections, or by combining the lines with binary features in order to measure the lengths of the segments.



All of the math routines make use of a previously stored second image (IP•2nd Image->Setup). The second image is never modified by these operations and will stay in effect until another image is selected and copied into memory. If the second image and the front image have different image modes (8 or 16 bit per channel depth, RGB color or Greyscale), the second image values are converted automatically. If the two images have different dimensions, the common rectangle in the upper left corner is processed.

-> Add

This routine adds together the pixel values in the frontmost image and the previously stored second image (which is not modified), and autoscales the result so that the smallest and largest values are black and white with no clipping.

-> Divide

This routine divides the pixel values in the previously stored second image (which is not modified) into those in the current image, and autoscales the result so that the smallest and largest values are black and white with no clipping. To prevent division by zero problems, a constant offset of 1.0 is added to the pixel values (0..255)

-> Multiply

This routine multiplies the pixel values in the previously stored second image (which is not modified) times the current image, and autoscales the result so that the smallest and largest values are black and white with no clipping.

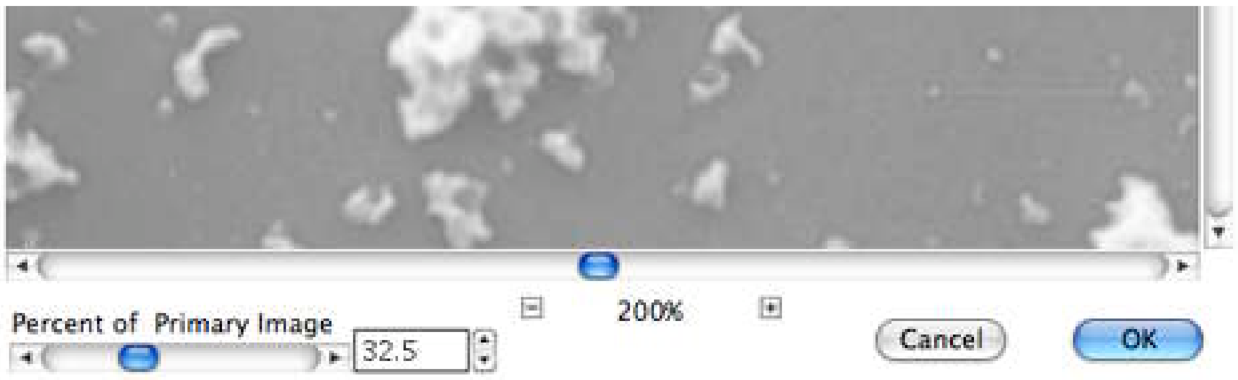
-> Subtract

This routine subtracts the pixel values in the previously stored second image (which is not modified) from the current image, and autoscales the result so that the smallest and largest values are black and white with no clipping.

-> Absolute Difference

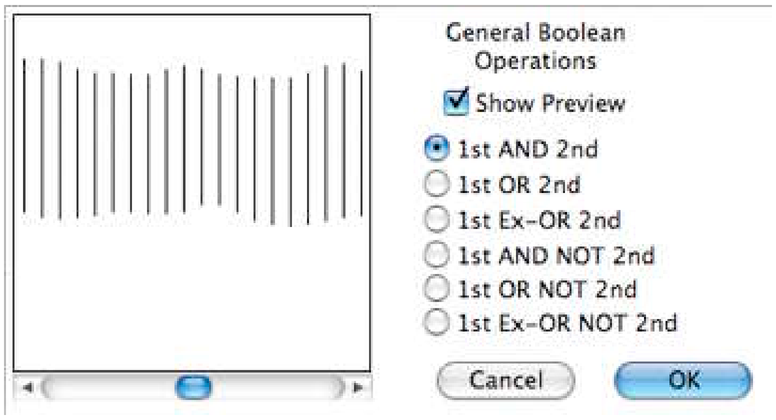
This routine calculates the absolute difference between the pixel values in the previously stored second image (which is not modified) and those in the current image, and autoscales the result so that the smallest and largest values are black and white with no clipping.

-> Blend with 2nd



This routine adds together the pixel values in the frontmost image and the previously stored second image (which is not modified), and autoscales the result so that the smallest and largest values are black and white with no clipping. It is identical to the -> Add function except that the percentage of the two images is adjustable by the user.

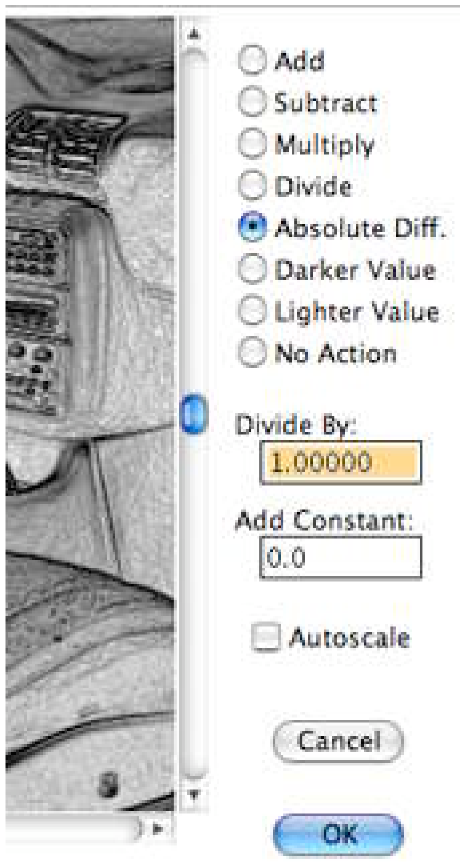
-> General Boolean



This routine allows combination of the previously stored second image with the current image using any of the Boolean operators (AND, OR, Exclusive-OR) plus the ability to invert the values in the second image (NOT). It is expected that the images will both be black and white binary images. White is interpreted as background or OFF, and any non-white pixel is considered to be ON.

-> **General Math**

This routine implements all of the arithmetic operations for combining two images, and thus duplicates the Add, Subtract, Multiply, Divide, Absolute Difference, Lighter and Darker routines listed separately. However, whereas they require no user dialog and always autoscale the resulting data, this plug-in allows the user complete flexibility in scaling the results by entering a multiplier and offset. This is typically useful when the pixel brightness values have been calibrated, e.g. for densitometry, when autoscaling would destroy the calibration. It is also possible to select "no action" and use the entered multiplier and offset on the image data.


Add
Subtract
Multiply
Divide
[✓ Absolute Diff.]
Darker Value
Lighter Value
No Action

Divide By: 1.00000

Add Constant: 0.0

☐ Autoscale

-> **Keep Darker Values**

This function combines the previously stored second image with the current image keeping at each pixel location whichever of the two original values is darker.

-> **Keep Lighter Values**

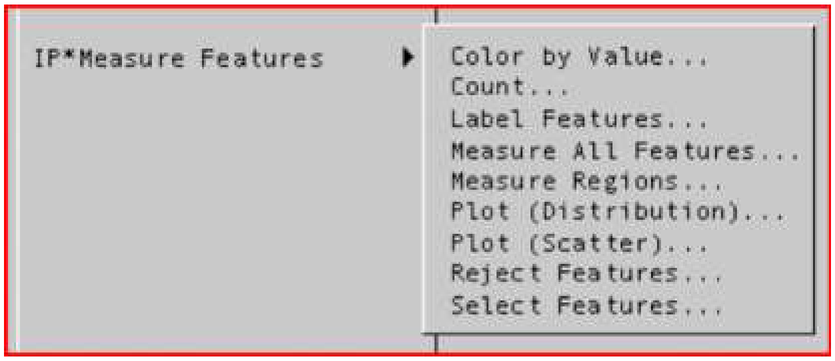
This function combines the previously stored second image with the current image keeping at each pixel location whichever of the two original values is lighter.

-> **Mask by 2nd**

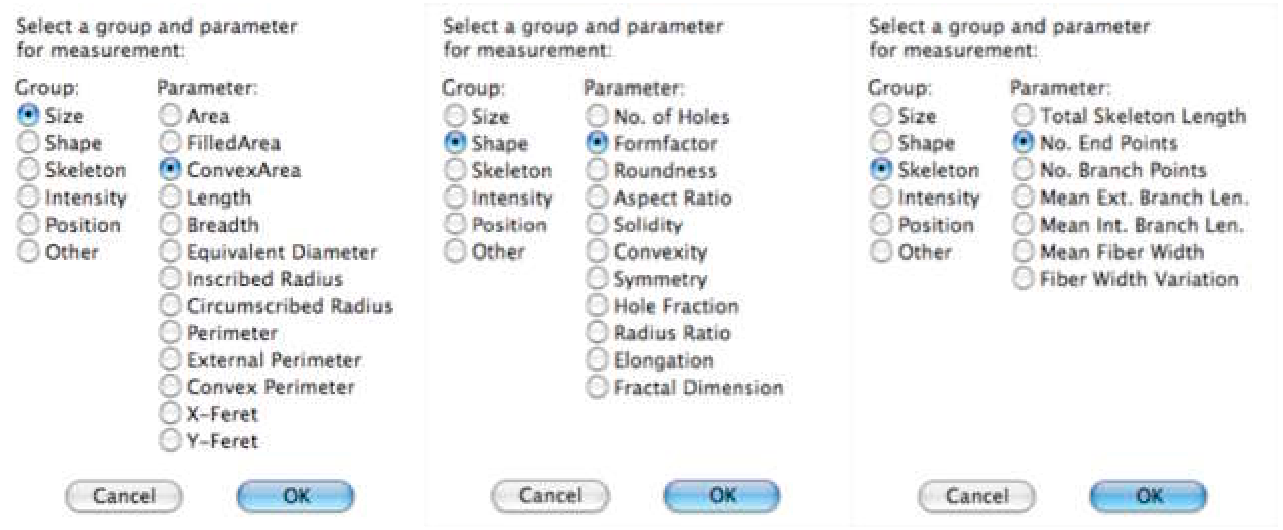
This function combines a previously stored binary image with the current image so that any pixel that is white (background) in the second image is erased to white in the current image, but any other pixels are left unchanged. This is particularly useful when a thresholded binary image has been processed (e.g., with a watershed segmentation) and is then combined as a mask with the original grey scale image so that the features are separated but retain their original pixel values.

-> Select Features by 2nd

The General Boolean routine listed above provides the AND, OR, ExOR and NOT functions for combining two images pixel-by-pixel. This routine provides a feature-based Boolean AND. It is different from the conventional Boolean operations in that it functions at the feature level. Also, unlike the General Boolean routine, the order of the images is important. One image, usually called the marker image, is placed beforehand into the second image memory (where it is not modified). Then this function will find any feature in the front image that has any pixel within it touching any of the markers in the second image. Features that do touch the second image markers are colored in the foreground color and those that do not are colored in the background color. The principal use of this function is to allow markers, including skeletons and outlines, to be used to select features for measurement. Both the marker image and the processed image are expected to be thresholded binary images, and any non-white pixel is treated as being part of a black feature.



Feature measurement operates on an image in which white pixels are interpreted as background and any non-edge-intersecting (except as noted) group of touching non-white pixels are interpreted as being a feature. If a non-rectangular area has been selected, this routine counts and measures features in the full bounding rectangle around the selection. Most of the feature measurement routines measure a common set of parameters on the features. Sometimes they are all measured and the data written to disk or used to generate graphics, sometimes the user is asked to select one or two parameters for measurement. The list of parameters is displayed in a hierarchical dialog shown below.


Select a group and parameter for measurement:

Group: [✓ Size] Shape Skeleton Intensity Position Other

Parameter: Area, FilledArea, [✓ ConvexArea], Length, Breadth, Equivalent Diameter, Inscribed Radius, Circumscribed Radius, Perimeter, External Perimeter, Convex Perimeter, X-Feret, Y-Feret

Select a group and parameter for measurement:

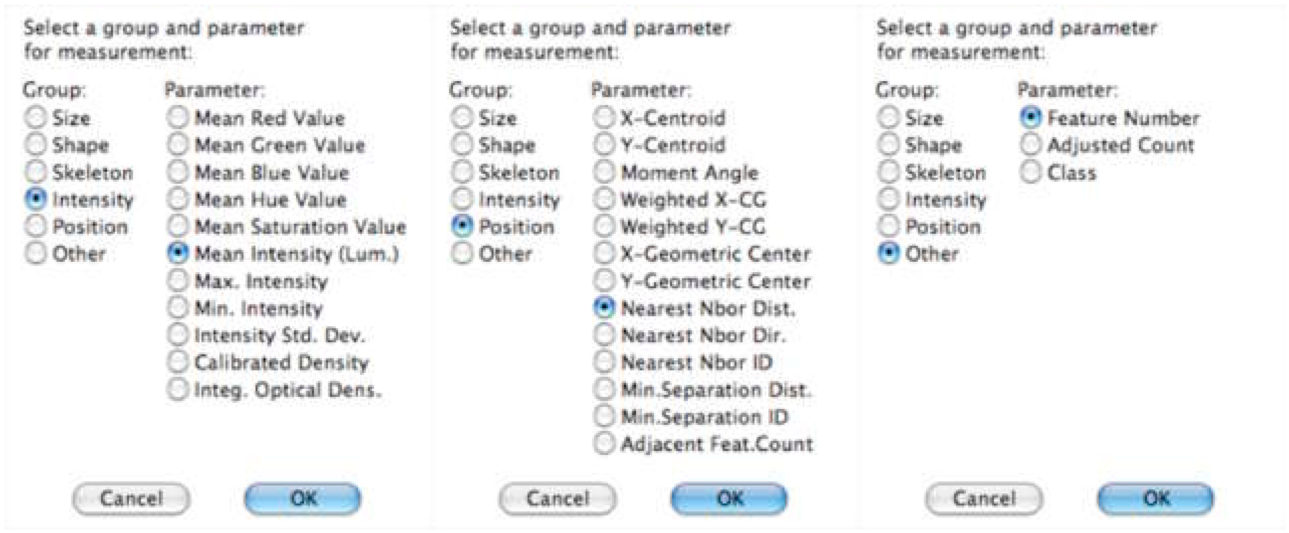
Group: Size [✓ Shape] Skeleton Intensity Position Other

Parameter: No. of Holes, [✓ Formfactor], Roundness, Aspect Ratio, Solidity, Convexity, Symmetry, Hole Fraction, Radius Ratio, Elongation, Fractal Dimension

Select a group and parameter for measurement:

Group: Size Shape [✓ Skeleton] Intensity Position Other

Parameter: Total Skeleton Length, [✓ No. End Points], No. Branch Points, Mean Ext. Branch Len., Mean Int. Branch Len., Mean Fiber Width, Fiber Width Variation



Select a group and parameter for measurement:

Group: Size, Shape, Skeleton, [✓ Intensity], Position, Other

Parameter: Mean Red Value, Mean Green Value, Mean Blue Value, Mean Hue Value, Mean Saturation Value, [✓ Mean Intensity (Lum.)], Max. Intensity, Min. Intensity, Intensity Std. Dev., Calibrated Density, Integ. Optical Dens.

Cancel OK

Select a group and parameter for measurement:

Group: Size, Shape, Skeleton, Intensity, [✓ Position], Other

Parameter: X-Centroid, Y-Centroid, Moment Angle, Weighted X-CG, Weighted Y-CG, X-Geometric Center, Y-Geometric Center, [✓ Nearest Nbor Dist.], Nearest Nbor Dir., Nearest Nbor ID, Min.Separation Dist., Min.Separation ID, Adjacent Feat.Count

Cancel OK

Select a group and parameter for measurement:

Group: Size, Shape, Skeleton, Intensity, Position, [✓ Other]

Parameter: [✓ Feature Number], Adjusted Count, Class

Cancel OK

-> Color By Value

This routine measures all of the features present and color codes them according to the magnitude of a selected parameter, with red being the smallest numeric value and proceeding through yellow, green, cyan and blue to magenta for the largest. If the image is in grey scale rather than color mode, the features will be assigned grey scale values that increase in brightness in proportion to the numeric value of the measurement.

-> Count

This routine simply counts the number of separate features present in the image that have the same color as the current pencolor. This makes it possible to use manual marking with the pencil tool in a unique color to indicate human-recognized features or stereological events, which are then counted (after which a different color can be used to repeat the operation for a different type of feature or event, and then finally the entire image can be saved with the marks as a record). This routine includes features that touch the edges, so the Reject Features function may be applied beforehand to erase features that touch two or four edges, depending on the purpose of the count.

-> Label Features

This function measures all of the features present and labels the image with the numeric value of the selected parameter for each feature. The labeling uses the current pencolor and erases a small surrounding rectangle to the current background color. Note that the labels are actually written into the pixels of the image, so it is usually wise to perform this operation on a duplicate image or a duplicate layer.

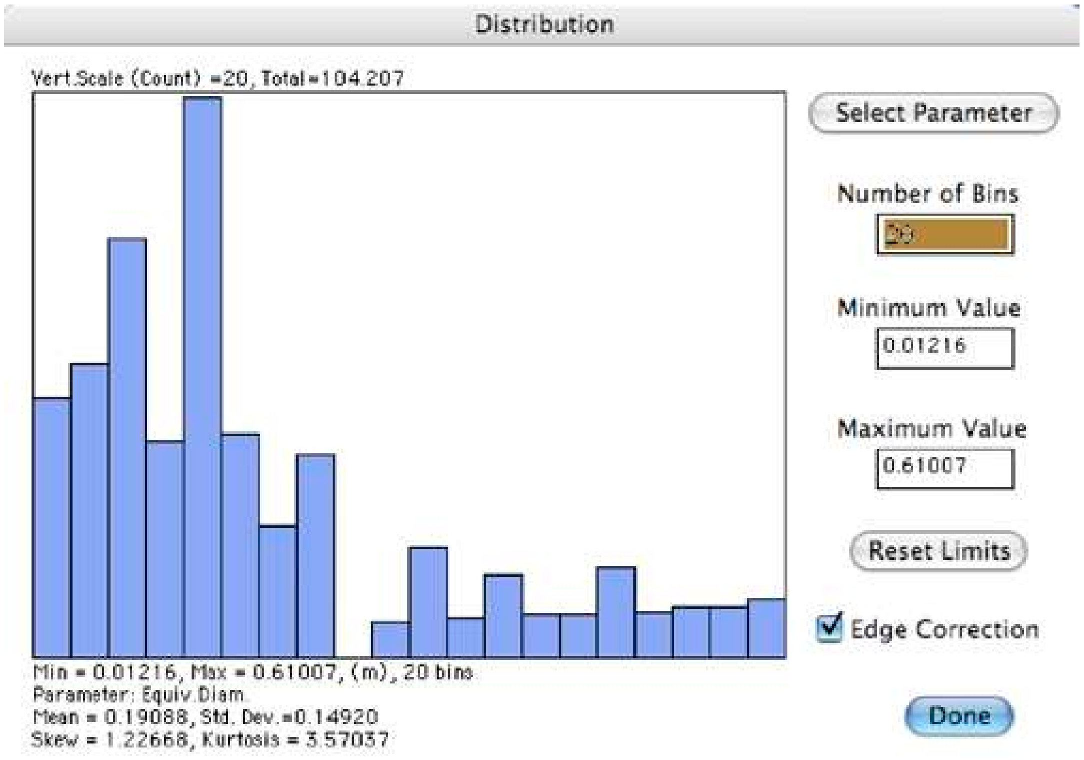
-> Measure All Features

This function measures all of the features for all of the parameters and writes the data to a disk file. The file is tab-delimited ascii text that can be read by most spreadsheet or data analysis programs. It is also possible to append the data to an existing file to build up a large data set from multiple images. In case some of the measured parameters are not of interest in a particular instance, the user can delete those columns of values (e.g. using an Excel macro).

-> Measure Regions

Unlike the other functions in this menu, which measure features as defined by consisting of contiguous non-white pixels, this function uses the region-of-interest outlines to define the features to be measured. The regions are actually held internally within Photoshop in an alpha channel, which can be displayed by converting the marquee outlines around the regions to a Quickmask mode that shows a colored overlay on the image. When the alpha channel is shown in this mode, it can be processed using any of the tools appropriate for binary images (e.g., morphology, watershed segmentation), and then converted back to the region outlines. Also unlike the other functions in this menu, the Measure Regions function will measure regions that abut to the edge of the image. The data are written to a file, exactly the same as for Measure All Features.

-> Plot (Distribution)



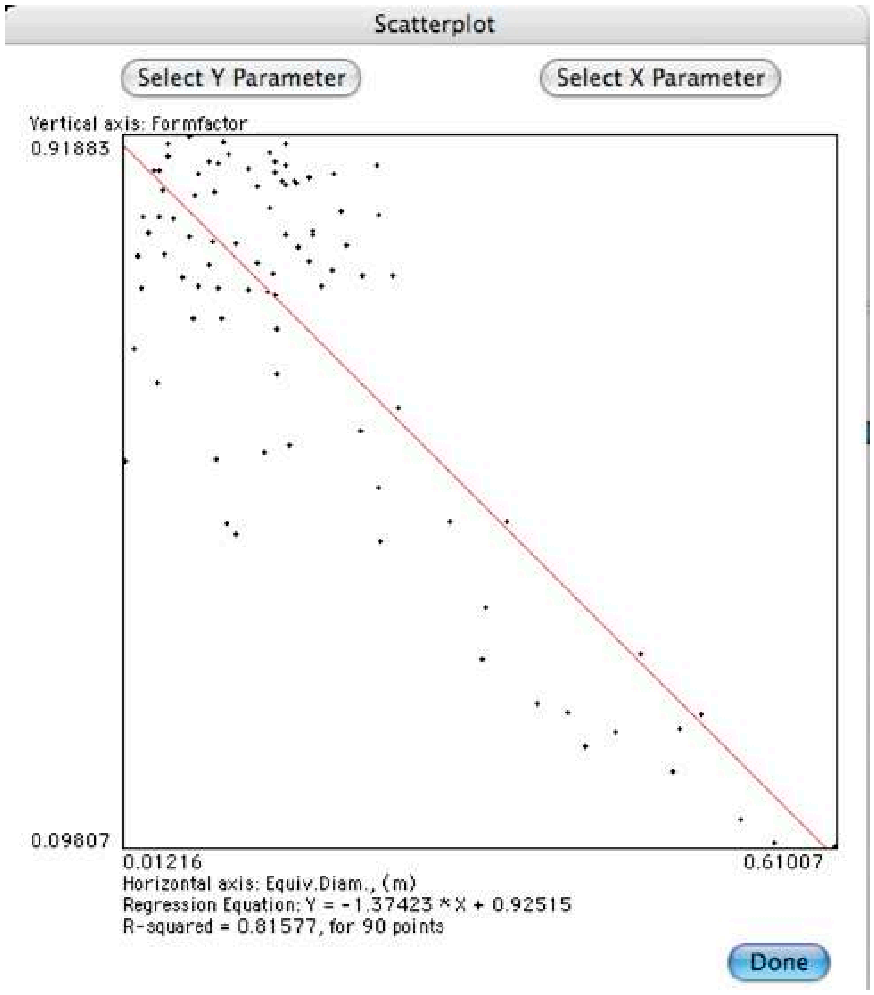
Vert.Scale (Count) =20, Total=104.207

Number of Bins: 20
Minimum Value: 0.01216
Maximum Value: 0.61007
[✓ Edge Correction]

Min = 0.01216, Max = 0.61007, (m), 20 bins
Parameter: Equiv.Diam
Mean = 0.19088, Std. Dev.=0.14920
Skew = 1.22668, Kurtosis = 3.57037

This function measures all of the features present for the selected parameter and displays the results in the form of a distribution or histogram plot. The default limits of the plot are the minimum and maximum measured values, but both these and the number of bins in the histogram (up to 30) can be set by the user. If the image contains all of the features in the population of interest within its borders then no edge adjustment is needed, but if the image is one field of view in a larger sample and some features intersect the edge of the image (so that they cannot be measured) then selecting edge correction instructs the program to make a statistical correction for the fact that large features are more likely to touch the edges than small ones. The graph also shows some simple statistics for the distribution. If logging is in effect when this routine is selected, the graph and data are saved to an html file that can be read with any browser.

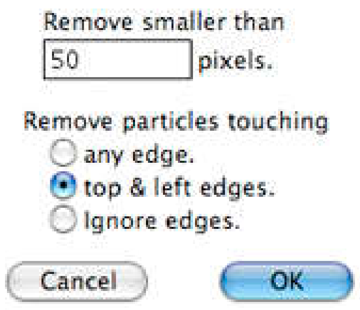
-> Plot (Scatter)



Vertical axis: Formfactor
0.91883
0.09807
0.01216
0.61007
Horizontal axis: Equiv.Diam., (m)
Regression Equation: Y = -1.37423 * X + 0.92515
R-squared = 0.81577, for 90 points

This function measures all of the features present for two selected parameters and displays the results as a scatterplot with each point representing the values for one feature. A best-fit regression line is also computed and the r-squared value indicating goodness of fit is shown. If logging is in effect when this routine is selected, the graph is saved to an html file that can be read with any browser.

-> Reject Features



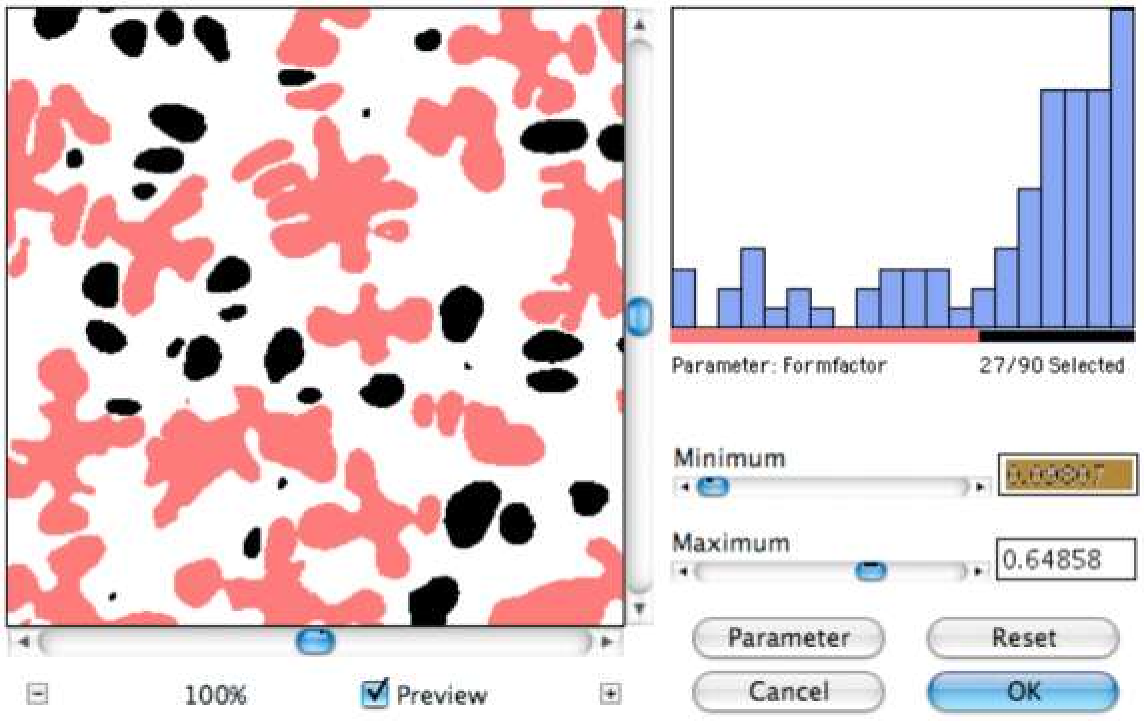
Remove smaller than 50 pixels.

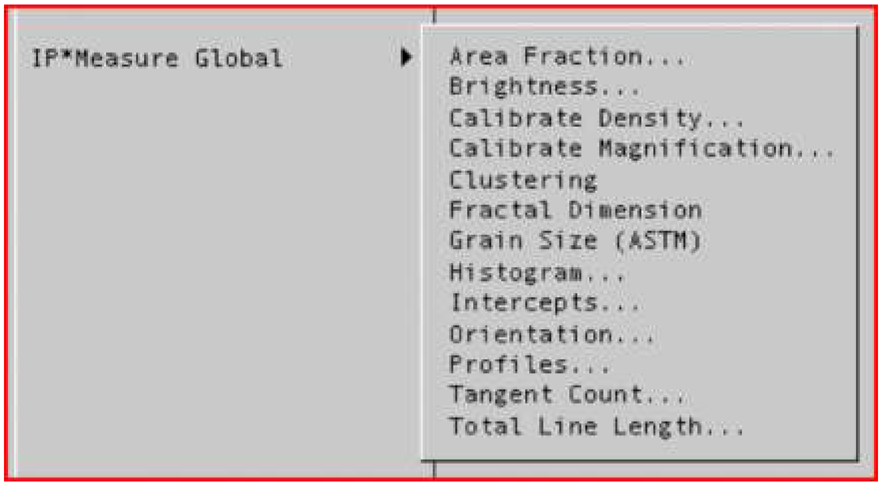
Remove particles touching
any edge.
[✓ top & left edges.]
Ignore edges.

This function is used to eliminate small or edge-touching features from a binary image before measurement. The user can enter a limit for feature size (area in pixels) to reject small bits of noise. It is also possible to eliminate features that touch any edge (and so cannot be measured) or that touch just two edges (which provides unbiased counting of number per unit area). The rejected features are colored in the background color, while those that are kept are colored in the foreground color. If these colors are the default black and white, this results in the erasure of the rejected features.

-> Select Features

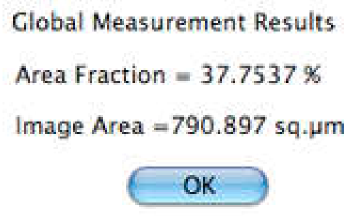
This function is used to select features based on the numeric values of a selected parameter. A distribution of the measured values is shown and limits can be entered to select the features to be kept. The selected features are colored in the foreground color, while those that are not selected are colored in the background color.





The functions in this group perform a variety of measurements on the entire image or on a selected region.

-> Area Fraction


Global Measurement Results

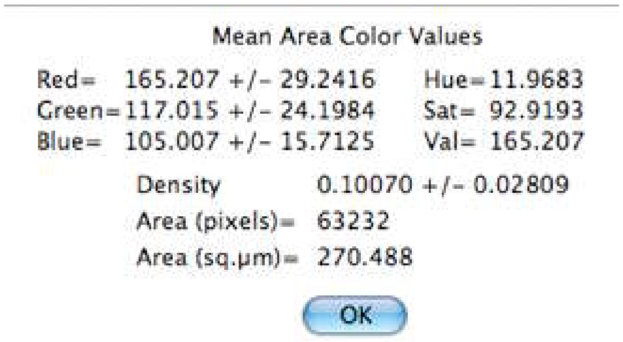
Area Fraction = 37.7537 %

Image Area =790.897 sq.µm

This function reports the area fraction of non-white pixels in the image, or the selected region. The area of the region is also reported using the established magnification calibration.

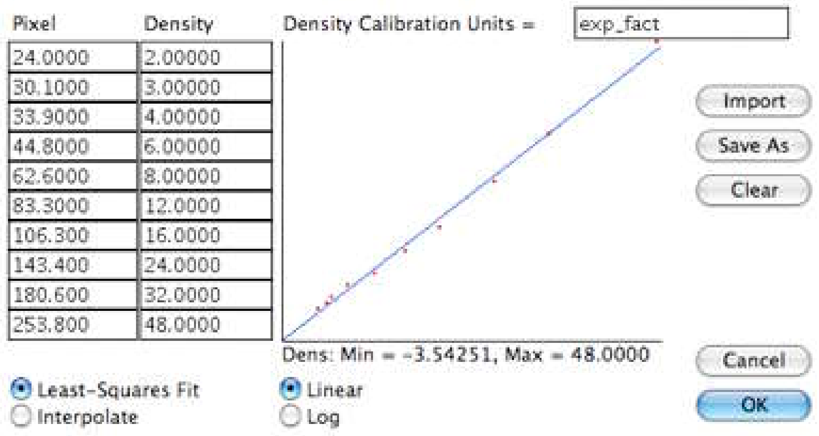
-> Brightness

This function reports the average brightness or color components of the pixels in the image, or the selected region. If a density calibration is in use, the calibrated result is also shown.


Mean Area Color Values

```
Red=   165.207 +/- 29.2416    Hue=11.9683
Green= 117.015 +/- 24.1984    Sat= 92.9193
Blue=  105.007 +/- 15.7125    Val= 165.207
    Density          0.10070 +/- 0.02809
    Area (pixels)=   63232
    Area (sq.µm)=    270.488
```

-> Calibrate Density


Density Calibration Units = exp_fact

Pixel	Density
24.0000	2.00000
30.1000	3.00000
33.9000	4.00000
44.8000	6.00000
62.6000	8.00000
83.3000	12.0000
106.300	16.0000
143.400	24.0000
180.600	32.0000
253.800	48.0000

Dens: Min = -3.54251, Max = 48.0000

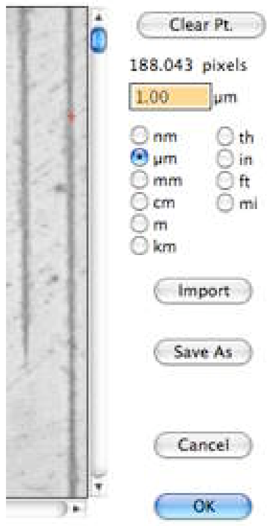
[✓ Least-Squares Fit] Interpolate

[✓ Linear] Log

This function allows the user to establish a calibration curve between pixel brightness (0..255 values are used for both 8 and 16 bit per channel images, the latter allowing for additional precision) and some meaningful parameter such as density. The typical procedure for establishing such a curve is to acquire images of standards, measure the brightness of the appropriate regions, and accumulate a list of at least 2 (up to 10) intensities for known standards. These values are entered into the table in the dialog presented by this plugin. A calibration curve can be fit through these points either by least squares fit or point-to-point interpolation, using either a straight line or log curve (which corresponds to Beer's law absorption). The resulting calibration curve can be stored on disk in a named text file for reloading and using as appropriate, and remains in effect until cleared or replaced by another calibration.

-> Calibrate Magnification

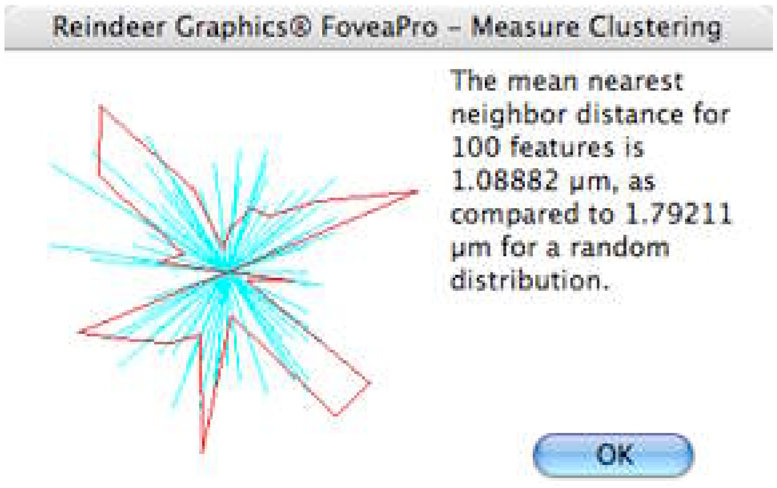
This function allows the user to establish a magnification calibration. The user is asked to mark two points on an image (usually an image of a known standard or scale), and to enter the distance between them (selecting appropriate units). This calibration can be saved to disk in a named text file, and recalled as appropriate. It is typical for a light microscope with fixed magnifications to store and recall calibrations for each objective lenses, for example. The magnification calibration is used for all measurements on all images, until it is changed. The magnification value is not saved with each image. Note that the magnification calibration used by the Fovea Pro plug-ins does not interact with the units used by Photoshop for measurements shown in the Info window, which is typically set to pixels but may also be in inches, etc.


188.043 pixels

1.00 µm

( ) nm [✓ µm] ( ) mm ( ) cm ( ) m ( ) km ( ) th ( ) in ( ) ft ( ) mi

-> Clustering


Reindeer Graphics® FoveaPro – Measure Clustering

The mean nearest neighbor distance for 100 features is 1.08882 µm, as compared to 1.79211 µm for a random distribution.

This routine is applied to an image of multiple features (non-white pixels) on a white background. The mean nearest neighbor distance (centroid-to-centroid) for the features is compared to the value for a random spatial distribution of the same number of features in the image area. If the observed mean is less than that for a random distribution it indicates clustering, while a greater value represents self-avoidance. The dialog also shows a rose plot of the nearest neighbor vectors, which can indicate nonuniformity and preferred orientation.

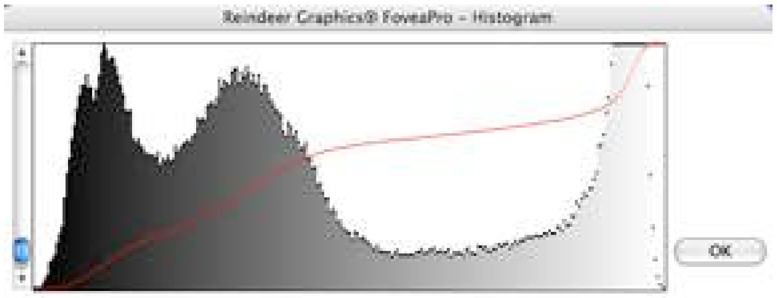
-> Fractal Dimension

This routine measures the fractal dimension of all of the boundaries in a binary image. It is not the same calculation as the feature-specific measurement that reports a value for each feature within the image, and is appropriate for dispersions of features, boundaries that extend across the image and off the edges, etc. (Placing a rectangular region around a specific feature can be used to measure just one feature.) The method used is based on the EDM, which is the most accurate procedure for fractal dimension measurement. (The feature-specific routine uses the slope of a Richardson plot for each feature.)

-> Grain Size (ASTM)

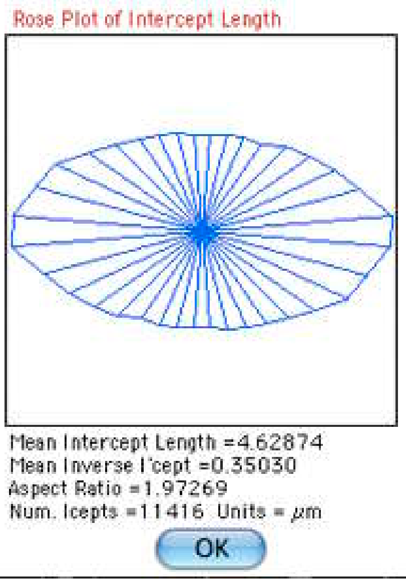
This routine is specifically intended for the measurement of metallic grains shown in a binary image. The result is reported on an arbitrary logarithmic scale that corresponds to the ASTM standard for grain size measurements, and has no units. The method employed is the linear intercept procedure, as specified in the ASTM standards. The image must be calibrated in inches, mm, cm or micrometers for the calculation to be performed. Note that the term "Grain Size," although in use for more than a century, is a misnomer, and that from a stereological point of view the measurement is not the size of the grains but rather is related to the grain boundary surface area per unit volume.

-> Histogram



This routine measures the brightness histogram for the image (or within a selected region). This is similar to the Photoshop histogram display but also saves the data to a tab-delimited text file that can be opened in a spreadsheet or other program for further analysis. The display allows the vertical scale to be changed, and can show either the brightness or RGB components for a color image. In addition to the conventional histogram plot (number of pixels as a function of brightness value), the display also shows the cumulative histogram. This is the integral of the histogram, which rises from 0 to 100%.

-> Intercepts

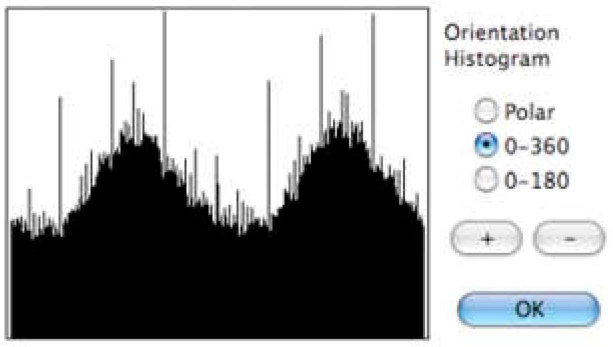


Mean Intercept Length =4.62874
Mean Inverse I'cept =0.35030
Aspect Ratio =1.97269
Num. Icepts =11416 Units = μm

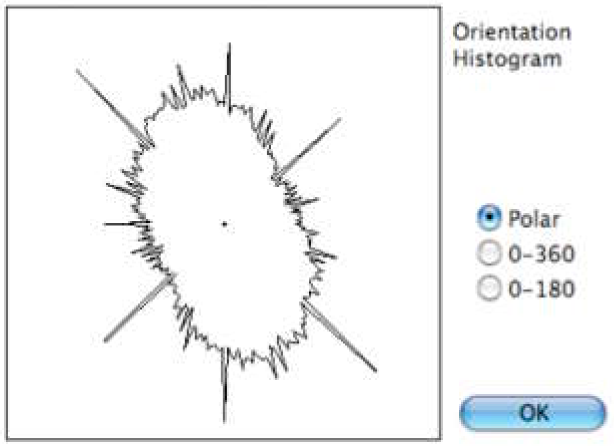
Several stereological procedures are based on the measurement of intercept lengths on binary images of structures. This routine performs an exhaustive procedure for determining intercept length information. A grid of parallel lines is ANDed with the image and the lengths of the individual intercepts measured, and this is repeated as the line grid is rotated in 10 degree steps. The mean intercept length and the mean inverse intercept length are reported in the calibrated magnification units, and the variation with angle is shown. The data are also saved in a tab-delimited text file for subsequent analysis.

-> Orientation

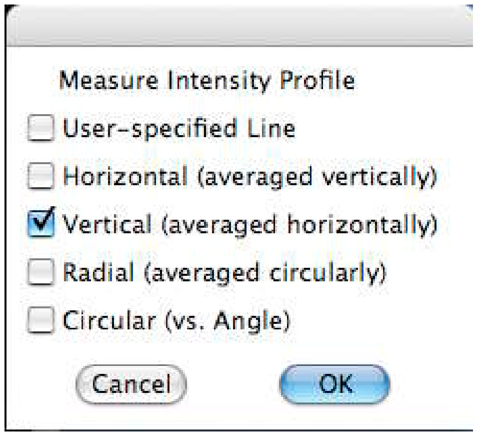
This function is intended to be used after the IP•Process->Orientation function has represented the orientation of the local brightness gradient by a grey scale value at each pixel location. It displays a histogram of those values and saves the data. The same data can be shown with the ->Histogram routine above, except that this routine also allows plotting the data combining the lower and upper halves of the histogram (to show values appropriate for the 0 to 180 degree range rather than 0 to 360 degrees), and to show the values as a rose plot. The “spikes” that occur at exact multiples of 45 degrees are an artefact of the edges of the square pixel grid and provide a useful built-in calibration scale.



!



-> Profiles



Measure Intensity Profile

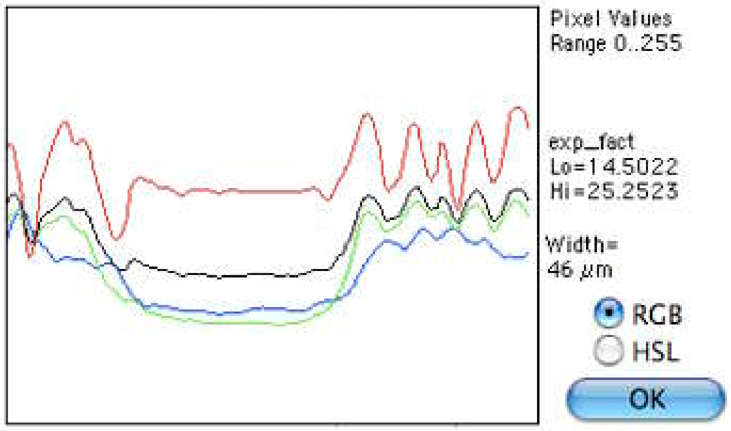
- [ ] User-specified Line
- [ ] Horizontal (averaged vertically)
- [✓] Vertical (averaged horizontally)
- [ ] Radial (averaged circularly)
- [ ] Circular (vs. Angle)

Cancel OK

There are five modes available for plotting intensity profiles from images. All of them save the data in a tab-delimited text file for subsequent analysis, and all of them plot the data both as raw intensity data and using any density calibration in effect.

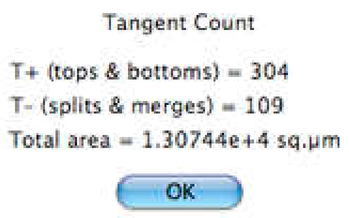
The Horizontal profile measurement is applied to a rectangular region, averages the pixel values vertically and plots the results as a function of horizontal position. The Vertical profile measurement routine is similar, but plots the results as a function of vertical position, with averaging in the horizontal direction. Both are useful for images of samples such as protein separations, X-ray diffraction films, cross-sections of layered structures, etc., in which there is one important direction of intensity variation. Measuring the intensity profile along a user-specified arbitrary line produces the same data format as the horizontal profile, but measures the intensity values along a straight line (or line segment) which the user selects by marking two end points on the image in a preview. The measured line is drawn onto the image afterwards for reference. By appending the data from this measurement to a previous measurement, it is possible to extend the measurement line and follow an irregular arbitrary path across the image.

The Radial profile routine measures the average radial intensity (not the color information) from the center of the image. The intensity values are averaged over all directions, so this routine is most suitable for images of structures that have radial symmetry (e.g., electron diffraction patterns, tree rings, etc.). In many cases the user can efficiently specify the location of the measurements by using the Photoshop rectangular or circular region selection, holding down the option/alt key to draw from the center, and the shift key to constrain the dimensions to a circle or square, and drawing a region starting at the center of the pattern. That center point will then become the origin for the radial measurement.



The Circular profile routine is similar but instead plots the intensity as a function of angle going around the circular or elliptical path specified by the region selection line (the pixels marked by the edge of the circular or elliptical path are measured; if a rectangular region is marked, the inscribed ellipse is measured). The angle range is 0-360 degrees, starting at the right side and proceeding counterclockwise (standard engineering coordinates). If the circle is large, many points will be averaged at each angle, while if it is very small the plot will show interpolation between the measured points.

-> Tangent Count



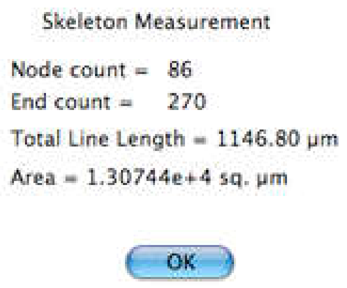
Tangent Count

T+ (tops & bottoms) = 304
T- (splits & merges) = 109
Total area = 1.30744e+4 sq.µm

This routine is applied to binary images of features (groups of non-white pixels on a white background) to determine the net tangent count. Positive (convex) and negative (concave) tangents are counted and the total value reported for stereological interpretation.

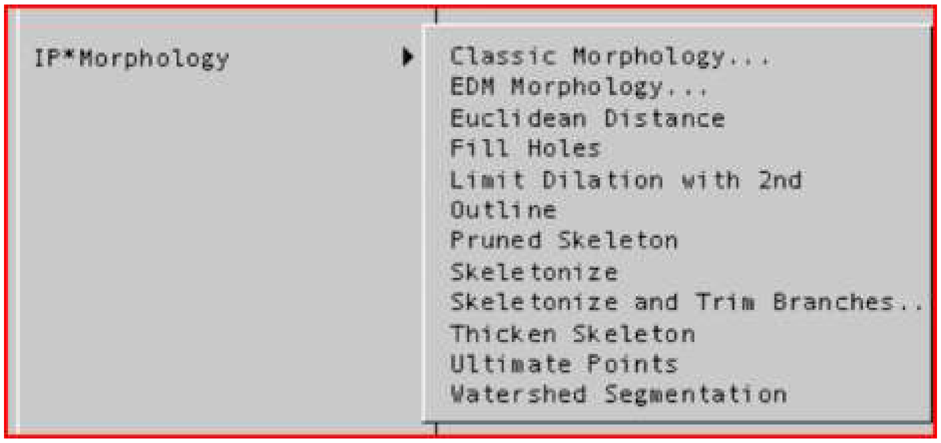
-> Total Line Length

This routine measures the total length of skeleton lines, or any other single-pixel wide lines (e.g., feature outlines produced by the IP•Morphology->Outlines routine). The measurement uses the same super-resolution procedure as the feature perimeter measurements. The reported value is given in the current calibrated magnification units. If the subject image contains a skeletonized result produced by the IP•Morphology->Skeletonize routine, the number of end and branch points is also shown.



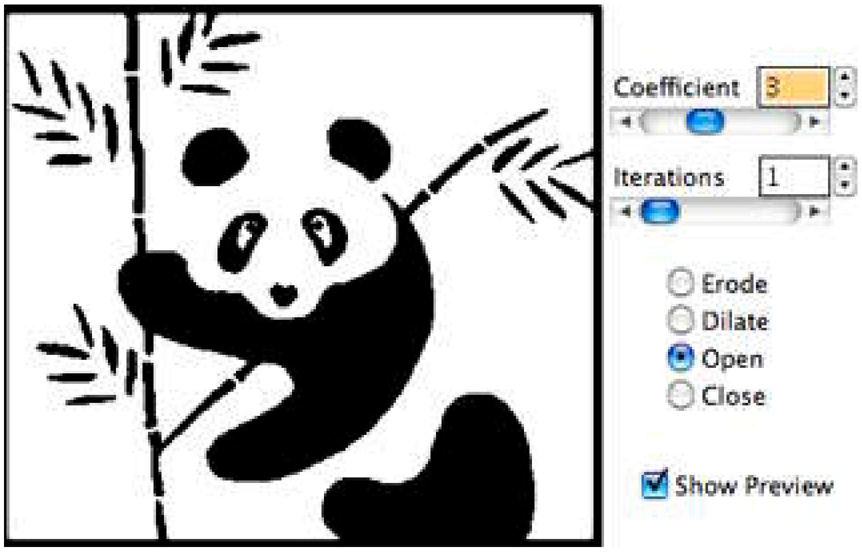
Skeleton Measurement

Node count = 86
End count = 270
Total Line Length = 1146.80 µm
Area = 1.30744e+4 sq. µm



This group of functions provide classical morphology routines (erosion, dilation, opening and closing, and related operations that consider each pixel's neighborhood). They should be applied to binary (black and white) images, and interpret any non-white pixel as part of a feature, and white pixels as background. Note that there are also morphological routines under the IP•Rank menu for dealing with color or grey scale images, which also offer erosion, dilation, opening and closing.

-> Classic Morphology



Coefficient: 3

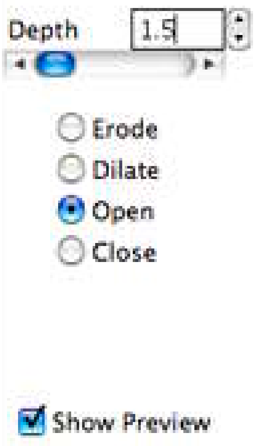
Iterations: 1

Erode [✓ Open] Dilate Close

[✓ Show Preview]

This routine performs classical erosion, dilation, opening (erosion followed by dilation) and closing (dilation followed by erosion). The number of iterations does not correspond exactly to the distance of the erosion, because classical procedures compare the pixel only to its immediate neighbors. The "Coefficient" value is the number of neighbor pixels that must be exceeded for the central pixel to change it color from black to white or vice versa. If this coefficient is set to zero, the result is traditional erosion or dilation (the pixel changes if it is adjacent to any opposite colored neighbor). Higher values provide more control over the process; a value of 7 (requiring all 8 neighbors to be opposite on color) results in the removal of only isolated single pixels. Note that Fovea Pro includes four sets of routines that perform morphological erosion, dilation, opening and closing. The EDM-based routines are more isotropic than classical methods. Grey scale and color erosion and dilation deal with pixels that have values other than black or white and are found under the IP•Rank menu.

-> EDM Morphology



Depth 1.5

Erode
Dilate
[✓ Open]
Close

[✓ Show Preview]

Fovea Pro includes four different methods for performing the morphological functions of erosion, dilation, opening and closing. Classical morphology applies to binary images but produces anisotropic results in which feature shapes change with repeated application. Grey scale and color morphology (under the IP•Rank menu) apply to images whose pixels have either brightness or color information rather than binary (black and white) definition of thresholded features. EDM-based morphology is applied to binary images (any non-white pixel is considered part of a feature), is much faster than classical morphology, and is isotropic, preserving shape information. The “Depth” distance entered into the dialog is not restricted to integer values, as for classical morphology, but can accept real numbers that correspond to actual pixel distances.

-> Euclidean Distance

This routine assigns the distance values as grey scale values for all of the non-white (feature) pixels in the image, representing the distance from that pixel to the nearest white (background) pixel. Because of the range limits of the image format, the maximum value is 255 pixels.

-> Fill Holes

This routine fills in all holes within features using the current foreground color. A hole is defined as white pixels that have no continuous path to any edge of the image. Background pixels are considered connected only if they share an edge (4-connectivity), whereas features (black pixels) are considered connected if they share either an edge or a corner (8-connectivity).

-> Limit Dilation with 2nd

This function performs classical dilation of features in the current image, but only within those regions that correspond to features in a previously stored second image. This procedure can be useful in dilating eroded seeds of features to reconstruct the features without merging. It can also be used as a way to accomplish marker selection of features (although it is less efficient than the IP•Math->Select Features by 2nd method).

-> Outline

This function erases the interior of features leaving just those edge pixels that are adjacent to (touch) the white background. The outlines are used for various stereological measurements, and also can be used as markers to select adjacent pixels or objects.

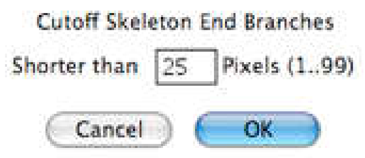
-> Pruned Skeleton

This series of menu items includes three skeletonization routines. All of them remove pixels from the edges of features leaving the midlines as 8-connected continuous lines. This routine then removes all branches that terminate within the image area, leaving only the lines that extend across the entire image and form a continuous tessellation. This is particularly appropriate for grain boundary or cell boundary images, and networks.

-> Skeletonize

This function thins the features to their midlines. It also marks the branch points (or nodes) and the end points of the skeleton so that they can be counted (see IP•Measure Global->Total Line Length) or selectively thresholded (see IP•Threshold->Select Skeleton Components).

-> Skeletonize and Trim Branches



Cutoff Skeleton End Branches

Shorter than 25 Pixels (1..99)

Cancel OK

This function performs the same skeletonization as the Skeletonize routine, followed by removing those terminal branches (branches that end within the image area) whose length is shorter than a user-entered value. The result of the function is to grey-code the short branches with a unique value so that the IP•Threshold->Select Skeleton Components function can keep or remove the short terminal branches.

-> Thicken Skeleton

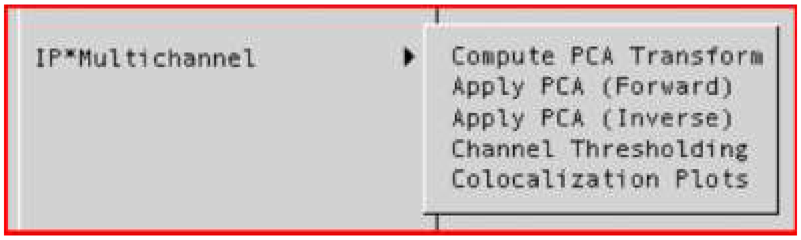
As noted above, the skeleton produced by all of the skeletonization procedures is an 8-connected line in which pixels are considered to be connected if they touch each other's sides or corners. This is the convention used for all feature connectivity. Background connectivity (as noted above) is 4-connected (pixels are connected if they touch along their sides, not their corners). This difference is inherent in a square pixel grid. If a skeleton is to be used subsequently to separate features (by removing it with a Boolean operation, or by inverting the skeleton image) it must be converted from 8-connected to 4-connected. This function accomplishes that conversion, by thickening the 8-connected skeleton to make it 4-connected.

-> Ultimate Points

The peaks in the EDM correspond to the centers of features that would be separated by the Watershed Segmentation routine, and their value of each corresponds to the radius of an inscribed circle within the feature. The ultimate points can be used to mark locations within features for various measurement procedures (unlike the feature centroids, which may lie outside of a concave shape, the ultimate point is always within the feature). Also, a histogram of the values of the ultimate points is a direct measurement of the number and size distribution of the features present.

-> Watershed Segmentation

This function generates and uses the EDM of a binary image to find the lines of separation between convex touching features and erases them so that the features are separated for measurement.



These routines operate on color (RGB, Lab, etc.) and multichannel images to perform principal components analysis. They are not included in the Image Processing Tool Kit. Note that multichannel images cannot be displayed in color except by selecting individual channels or converting them to standard color images, which may hide some of the channels. Before using the PCA transformations it is necessary to select all of the channels of interest in the channels window. This is facilitated by a “script” that should be installed in Photoshop (place the file “IP_SelectAllChannels.js” in the Adobe Photoshop CS2 / Presets / Scripts). Select this function as File-> Scripts -> IP_SelectAllChannels. The script (written in Java) selects and turns on for display all channels in a multichannel image.

-> Compute PCA Transform

After selecting some or all of the multiple channels to be used in the PCA procedure, this function performs the computation of the eigenmatrix, which is stored on disk (file tupe *.pca). The covariance matrix, including the significance of each new channel, is also displayed and written to the html log file if one is in use.

-> Apply PCA (Forward)

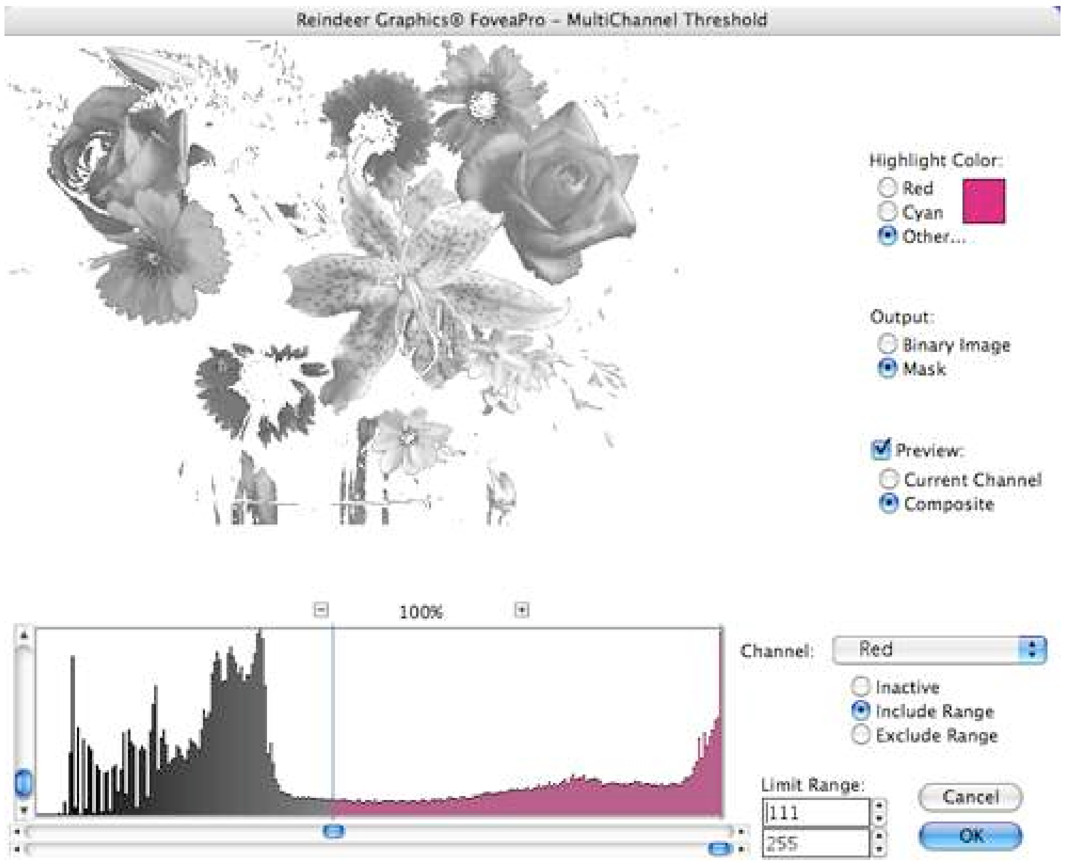
This routine applies a previously calculated Principal Components Analysis eigenmatrix to the multichannel image, transforming the current channels to values along the principle axes. The file for the matrix can be selected by the user. The pixel values in the channels are remapped according to the transformed axes, autoscaled to the range of the actual data in the image. For a standard color image (RGB, Lab, CMYK) this results in a display in which the first channel has the most significant variation, the second has the next most significant, and so on.

-> Apply PCA (Inverse)

This routine applies a previously calculated PCA eigenmatrix to a previously transformed multichannel image, to recover the original channel values. In most cases the transformed channels will have been modified (e.g., by noise reduction) or one or more will be eliminated e.g., by filling the channel with an average value) so that the recovered channels will be altered.

-> Channel Thresholding

Thresholding of multichannel images (including RGB images as well as multichannel images that have been transformed using PCA) can be performed using this function. The dialog allows setting upper and lower threshold levels on each channel, either to include or exclude pixels that lie within the selected intensity range. The preview can show either the results of the selection on the current channel or an AND that applies all of the active channel selections. The final thresholded result combines all active channel selections to produce a binary image of the selected pixels.


Reindeer Graphics® FoveaPro – MultiChannel Threshold

Highlight Color: Red [✓ Other...] Cyan

Output: Binary Image [✓ Mask]

[✓ Preview:] Current Channel [✓ Composite]

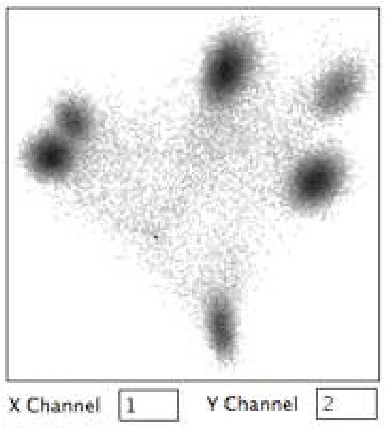
100%

Channel: Red

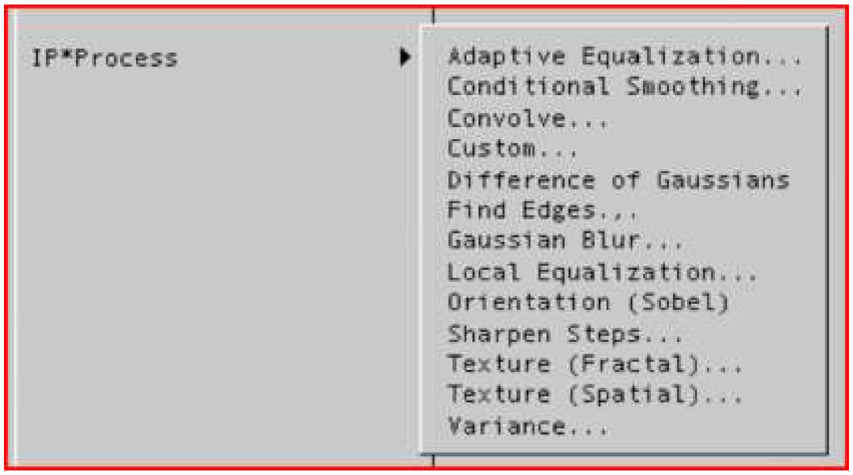
Inactive [✓ Include Range] Exclude Range

Limit Range: 111 255

-> Colocalization plots

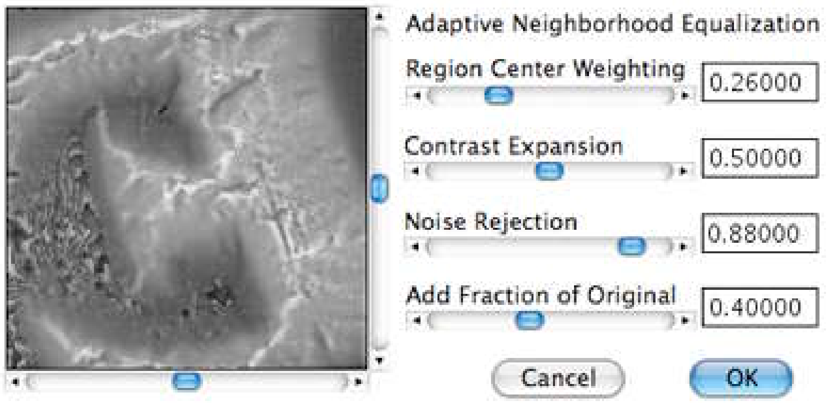

X Channel 1 Y Channel 2

Used after performing a principal components transformation on an image, this function displays plots of the distributions of pixel values along the new transformed principal axes. The channels are numbered sequentially in order of significance, although the names still correspond to the original axes unless they are changed by the user. This function can also be applied to multichannel images that have not been transformed by principal components analysis, for example to compare the localization of red, green and blue values.



The processing routines operate on grey scale or color images and produce a modified grey scale or color image in which some details (e.g., random noise) are suppressed while others (e.g., edges) are enhanced. Some of these routines have more than one use, depending on the nature of the image and the size of the neighborhood. Most of them operate on color images by internally converting the RGB values to HSI space and process the intensity values, leaving color information unchanged (exceptions are noted).

-> Adaptive Equalization


Adaptive Neighborhood Equalization

Region Center Weighting 0.26000

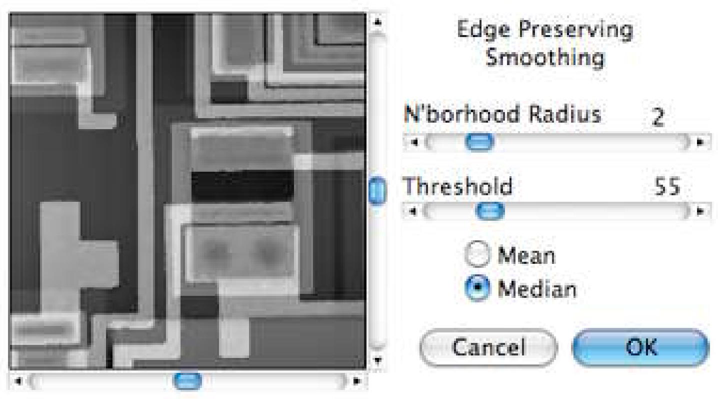
Contrast Expansion 0.50000

Noise Rejection 0.88000

Add Fraction of Original 0.40000

This function is a more general version of the ->Local Equalization routine, below. In addition to the basic procedure of comparing each pixel to its neighborhood and making pixels darker than their surroundings darker still (and vice versa), this routine weights the neighborhood pixels in importance as a way to adapt to the local contrast, and has a user-adjustable control to non-linearly increase the amount of local contrast added. Noise typically frustrates efforts at local detail enhancement, and this routine has provision for noise suppression.

-> Conditional Smoothing


Edge Preserving Smoothing

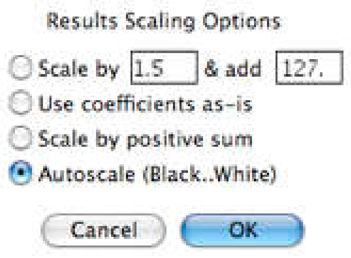
N'borhood Radius 2

Threshold 55

Mean [✓ Median]

Classical methods for noise reduction include neighborhood averaging and median filtering within a user-defined neighborhood size. This routine modifies the neighborhood by excluding pixels that are very different in brightness from the central pixel, based on a user-entered threshold. This prevents the neighborhood from crossing boundaries, keeping the boundary sharpness and preserving fine lines.

-> Convolve

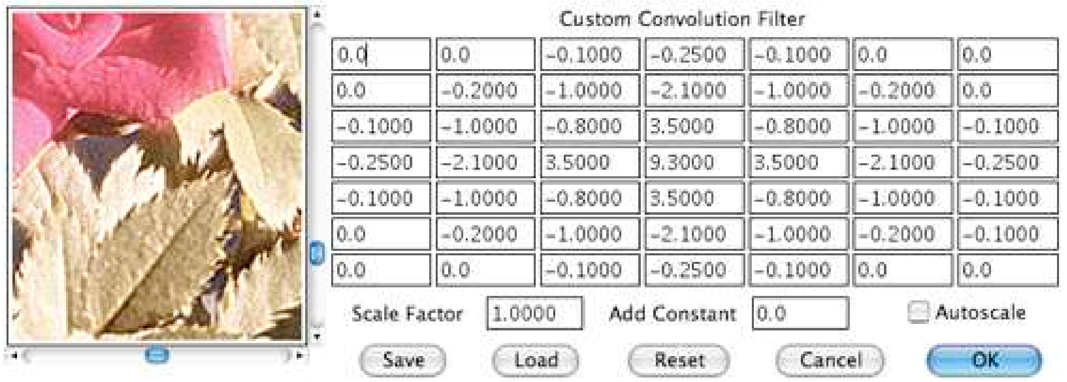

Results Scaling Options

- Scale by 1.5 & add 127.
- Use coefficients as-is
- Scale by positive sum
- [✓ Autoscale (Black..White)]

This routine operates like the ->Custom routine using a stored array of weight values (either integer or floating point values). The array must be a tab or space delimited file with a square array having an odd dimension, up to 47 x 47. The user can choose various ways to adjust the resulting product to keep the numerical results of the procedure within the 0..255 range without clipping.

-> Custom

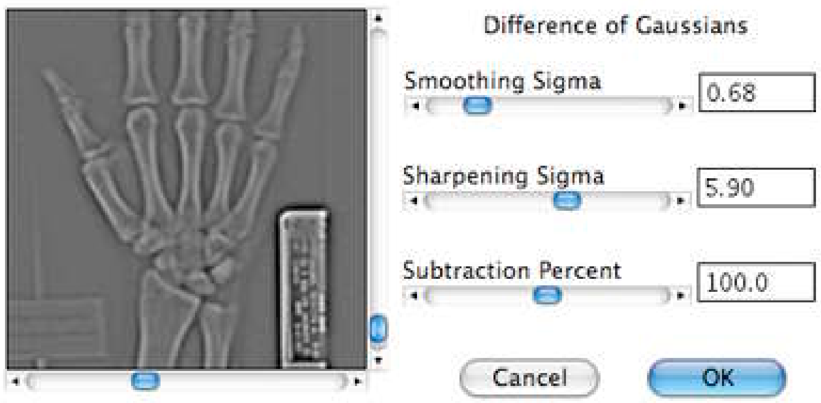
This routine is similar to the built-in Photoshop Filter->Custom routine that applies a kernel or array of weight values to an image, which can be used for noise reduction, edge delineation, embossing, and many other purposes. It can, in fact, import Photoshop kernel files (*.acf files), as well as loading kernels in the tab- or space-delimited text format supported by the Convolve routine. The important differences are 1) the array is larger (7x7 vs 5x5); 2) the weight values as well as the scale multiplier and offset value can be real numbers rather than being limited to integers; 3) the calculation is applied to the intensity channel of color images, leaving the color information unchanged (the Photoshop routine is applied separately to the RGB channels which can produce color shifts); 4) the result of the calculation can optionally be autoscaled so that the data fit exactly into the 0..255 range with no clipping of values occurs. Note: for Photoshop versions 7.0 and prior, the built-in Custom routine operates only on 8 bit images, but like all Fovea Pro plug-ins, this routine works on 16 bit per channel images in Photoshop 4 and later.


Custom Convolution Filter

0.0	0.0	-0.1000	-0.2500	-0.1000	0.0	0.0
0.0	-0.2000	-1.0000	-2.1000	-1.0000	-0.2000	0.0
-0.1000	-1.0000	-0.8000	3.5000	-0.8000	-1.0000	-0.1000
-0.2500	-2.1000	3.5000	9.3000	3.5000	-2.1000	-0.2500
-0.1000	-1.0000	-0.8000	3.5000	-0.8000	-1.0000	-0.1000
0.0	-0.2000	-1.0000	-2.1000	-1.0000	-0.2000	-0.1000
0.0	0.0	-0.1000	-0.2500	-0.1000	0.0	0.0

Scale Factor 1.0000 Add Constant 0.0 ☐ Autoscale

-> Difference of Gaussians


Difference of Gaussians

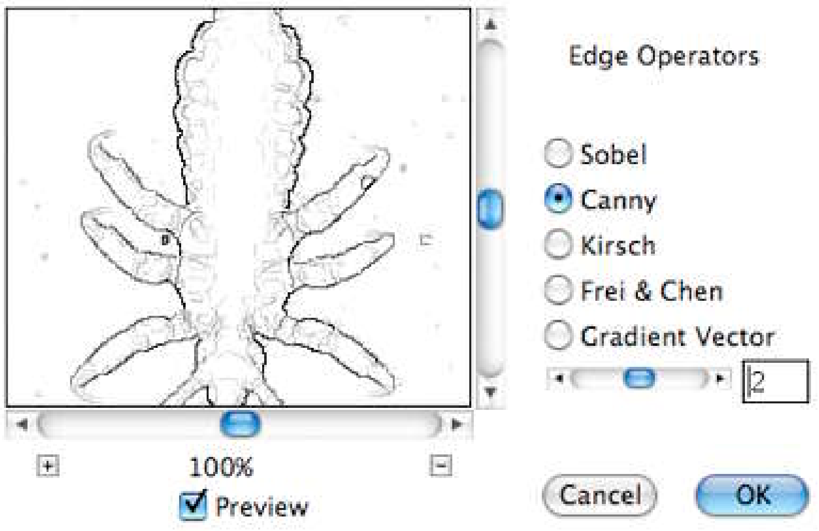
Smoothing Sigma: 0.68

Sharpening Sigma: 5.90

Subtraction Percent: 100.0

This routine is primarily used as a detail and edge enhancement tool for noisy images. Based on principles of human vision, it calculates the difference between two copies of the image, one smoothed by a small amount to remove the random noise, and the second smoothed by a larger amount to remove the important detail. This is equivalent to a band-pass filter performed in Fourier transform space. Typically the ratio of the two smoothing radii is between 3:1 and 8:1. The user can also determine the amount of the subtraction (the classical difference of Gaussians corresponds to 100%, after which the result may be added back to the original image).

-> Find Edges


Edge Operators

Sobel [✓ Canny] Kirsch Frei & Chen Gradient Vector

2

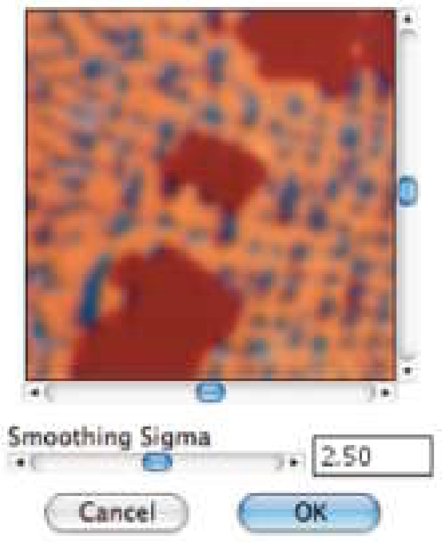
100%

[✓ Preview]

There are a great many edge-finding algorithms used in image processing. This routine implements several of the more common and a few of the less common but very powerful methods. The dialog allows selection of the method with a preview. The Sobel is the magnitude of the brightness gradient vector based on a 3x3 neighborhood; the gradient vector method is similar but allows the user to adjust the neighborhood radius. The Kirsch method is an older technique that finds the maximum of the derivative in eight directions. The Canny uses a gradient vector method but refines the result to keep just the local maxima to define the precise edge location. The Frei and Chen technique applies 7 local kernels to each pixel to detect edges with less sensitivity to noise and to the absolute brightness values. All of these except the basic Sobel operate on the intensity information without affecting colors. The Sobel operates on the individual RGB channels, similar to the Photoshop built-in Find Edges filter.

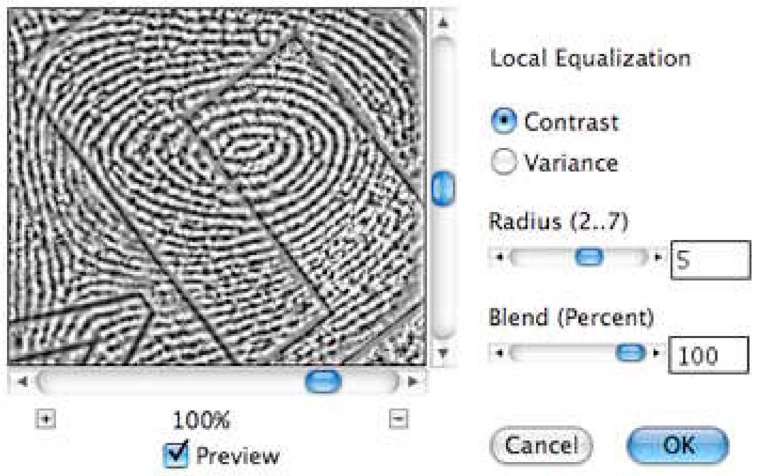
-> Gaussian Blur

This routine is similar to the built-in Photoshop Filter->Smooth->Gaussian blur routine and has similar uses. The difference is that the processing is applied to the intensity channel without affecting color information, while the Photoshop routine operates on the individual RGB channels. Also, in Photoshop versions prior to 7.0, the built-in function worked only for 8 bit per channel images, but like all Fovea Pro plug-ins, this routine works on 16 bit per channel images in Photoshop 4 and later.



Smoothing Sigma 2.50

-> Local Equalization



Local Equalization

Contrast [✓]
Variance

Radius (2..7) 5

Blend (Percent) 100

100%
Preview [✓]

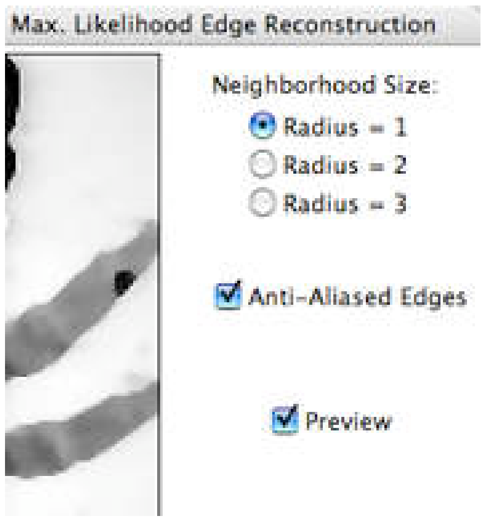
This function performs classical local histogram equalization within a circular neighborhood with user-defined radius, but keeps the new pixel brightness value only for the central pixel. This makes pixels slightly darker than their neighbors darker still and vice versa, increasing contrast for local detail. Alternately, the routine can equalize the local pixel variance, which produces similar results but is somewhat more resistant to random noise. The user can blend in a proportion of the original image with the equalized result.

-> Orientation

This function uses the Sobel local gradient method but rather than saving the magnitude of the gradient vector (as in ->Find Edges), it keeps the angle. The angle values are scaled to the 0..255 range, so that one step in brightness corresponds to an angle of about 1.4 degrees. This method is commonly used to measure the orientation of structure, and a histogram of orientations in the resulting image may be measured with the IP•Measure Global->Orientation routine. It is also sometimes useful to color code the orientation values using a spectrum of hues.

-> Sharpen Steps

This function modifies pixel values adjacent to steps so that the steps become more abrupt and hence easier to visualize and locate. The underlying method is a maximum likelihood reconstruction using statistical tests in a user-adjustable neighborhood. It is also possible to select a post-processing step that anti-aliases the step edges by smoothing in a direction parallel to the local edge orientation.

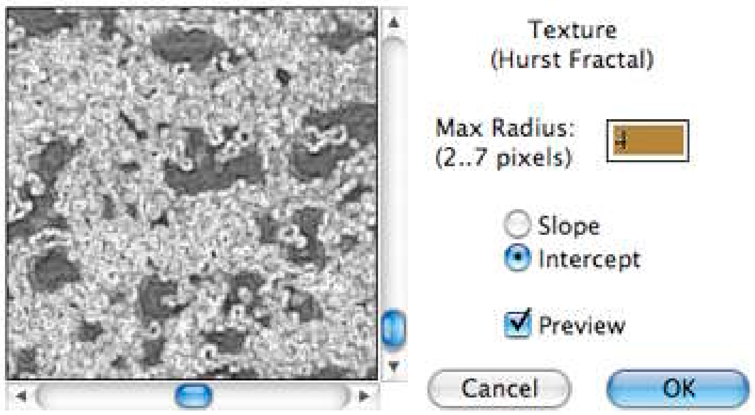


Neighborhood Size: [✓ Radius = 1] Radius = 2 Radius = 3

Anti-Aliased Edges [✓]

Preview [✓]

-> Texture (Fractal)



Max Radius: (2..7 pixels) 4

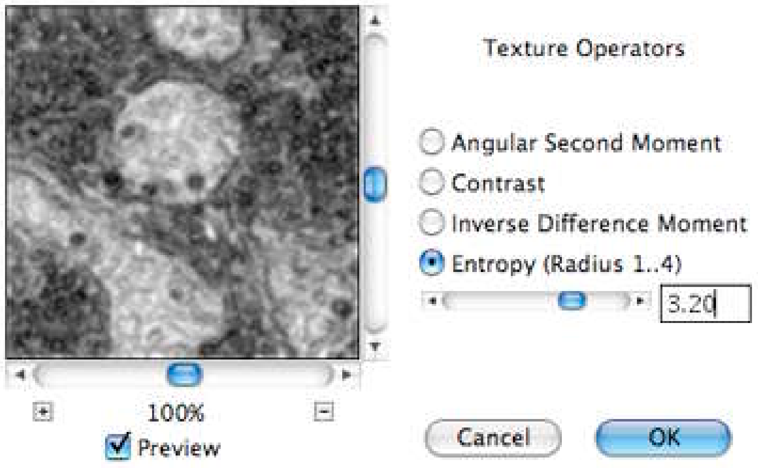
Slope [✓ Intercept]

Preview [✓]

There are two procedures that detect textures in images and convert them to grey scale differences to enable thresholding. This routine calculates the local fractal dimension of the brightness values by constructing a plot (on log-log axes) of the brightness range as a function of radius. The user can adjust the maximum radius, and select whether to use the slope or the intercept of the plot.

-> Texture (Spatial)

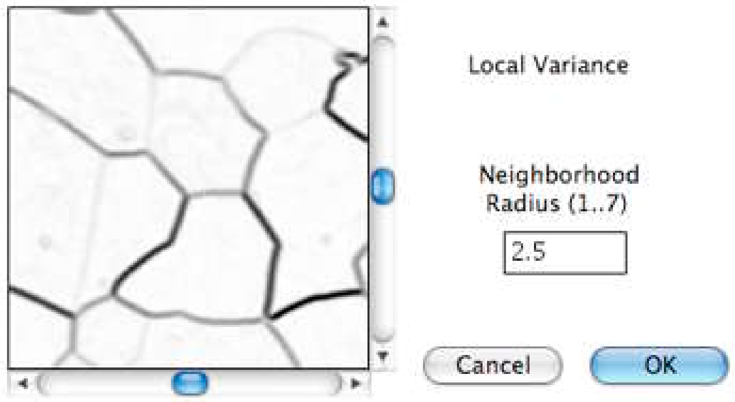
This texture extraction routine operates on the pixel values using a co-occurrence matrix. The terminology for the various methods follows the Haralick definitions. For the entropy calculation, the size of the neighborhood is user-adjustable.



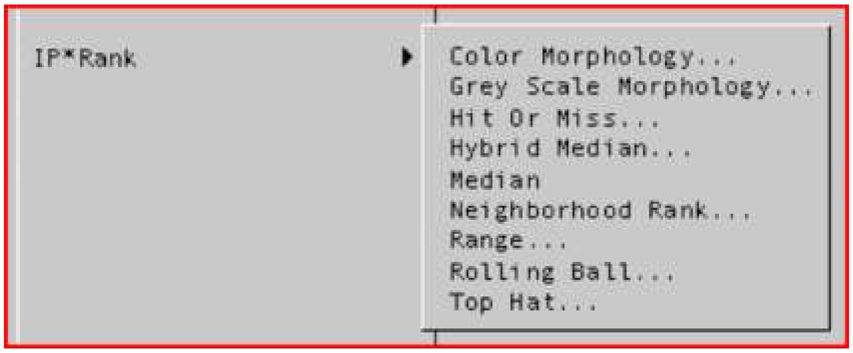
Angular Second Moment Contrast Inverse Difference Moment [✓ Entropy (Radius 1..4)] 3.20

Preview [✓]

-> Variance

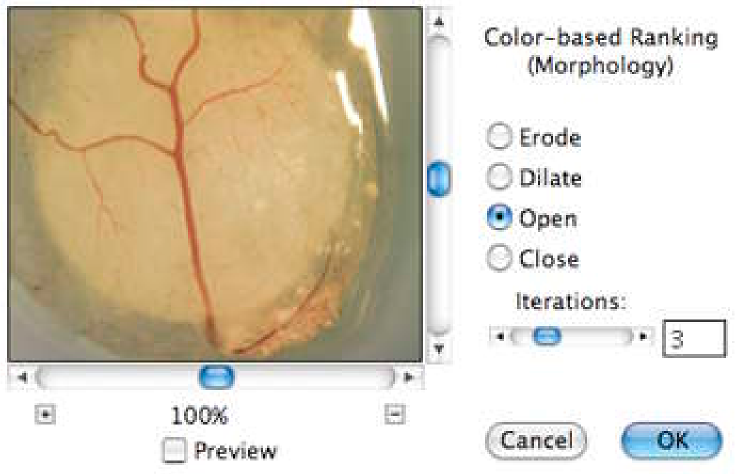


This procedure calculates the statistical variance of pixel brightness values in the user-adjustable neighborhood. This technique can be used for either edge or texture detection depending on the image characteristics and neighborhood size.



All of the routines in this group operate by ranking the pixels in a local neighborhood, after which various results may be selected such as brightest, darkest, median, etc. Most of them operate (except for the color morphology routine) on grey scale brightness values, and for color images will process the intensity while preserving color information. If it is desired to operate on the individual RGB channels, they can be selected in the Photoshop Channels window and processed individually.

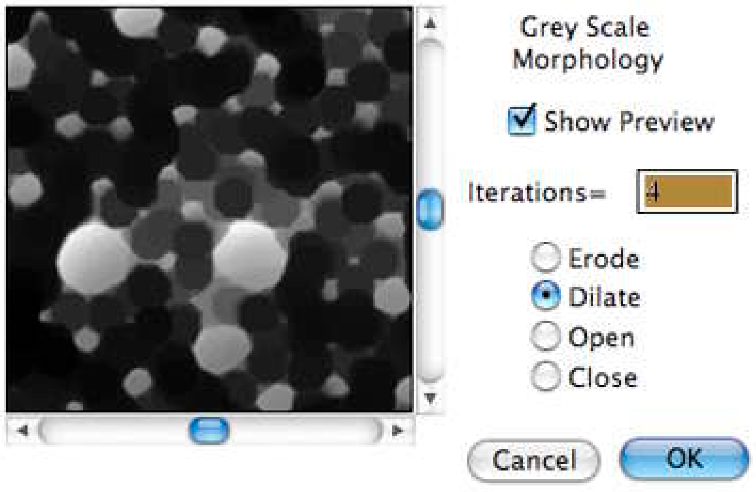
-> Color Morphology



This routine performs morphological operations on color images. The user should select the color of interest beforehand using the eyedropper. The plug-in then ranks the pixels in the neighborhood in terms of their similarity to the selected color. Dilation is accomplished by keeping the neighboring pixel value most like the pencolor, while erosion keeps the value most different from the pencolor. Opening is an erosion followed by a dilation, and closing is the opposite. The number of iterations can also be adjusted.

-> Grey Scale Morphology

This routine performs morphological operations on grey scale images (applied to a binary image, it is equivalent to classical morphology with a coefficient of zero; applied to a color image it is equivalent to the color morphology plug-in using black as the color of interest). The plug-in ranks the pixels in the neighborhood in terms of their brightness. Dilation is accomplished by keeping the darkest pixel value, while erosion keeps the brightest value. Opening is an erosion followed by a dilation, and closing is the opposite. The number of iterations can also be adjusted.

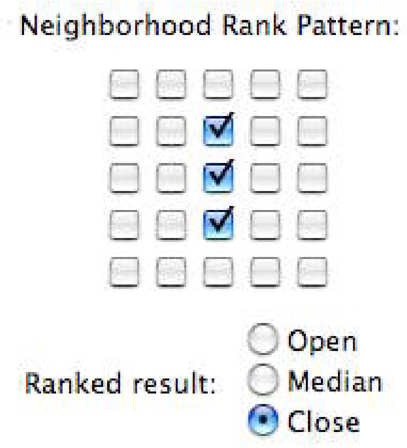

Grey Scale Morphology

[✓ Show Preview]

Iterations= 4

Erode [✓ Dilate] Open Close

-> Hit or Miss


Neighborhood Rank Pattern:

Ranked result: Open Median [✓ Close]

Most ranking operations function with a circular neighborhood in order to achieve isotropic results (unlike the Photoshop Filter->Noise->Median, Filter->Other->Minimum and ->Maximum routines, which use a square neighborhood). Sometimes it is advantageous to design a specific neighborhood shape if the nature of the image and its defects is known (e.g., scratches in a particular direction). This function allows the user to specify the neighborhood positions that are to be included in the ranking procedure, after which the median value can be selected, or the result of an opening (lightest followed by darkest) or closing (darkest followed by lightest).

-> Hybrid Median

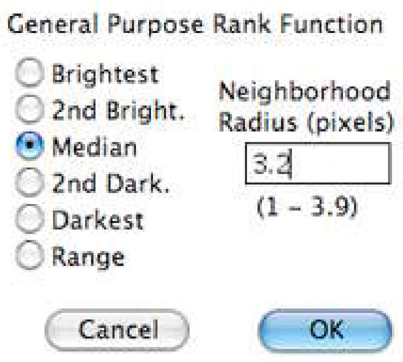
The classical median filter is a very widely used and effective noise reduction filter, and does not shift edges. However, it can round corners or eliminate narrow lines. The hybrid median preserves corners and lines by performing multiple rankings in different directions at each position. The user can select the size of the neighborhood, up to a radius of 6 pixels (at larger sizes, the preview may be slow to update).

-> Median

This routine is like the built-in Photoshop Filter->Noise->Median filter except that it uses a neighborhood that is a 5 pixel wide circle (the Photoshop routine uses a square neighborhood).

-> Neighborhood Rank

This routine performs ranking in a user-adjustable circular neighborhood, after which various pixel values (minimum, maximum, median, etc.) can be kept. The built-in Photoshop Median, Minimum and Maximum routines use square neighborhoods and limit the size selection to integers.

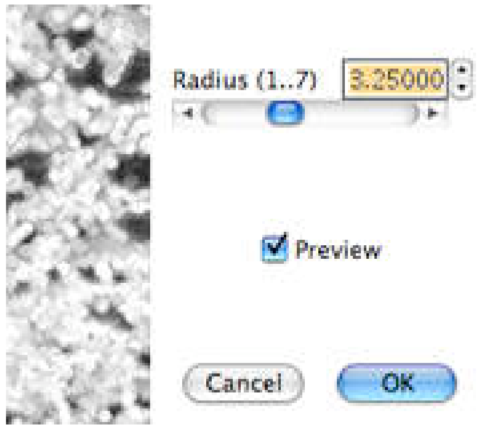

General Purpose Rank Function

Brightest [ ] 2nd Bright. [ ] [✓ Median] 2nd Dark. [ ] Darkest [ ] Range [ ]

Neighborhood Radius (pixels): 3.2 (1 – 3.9)

-> Range

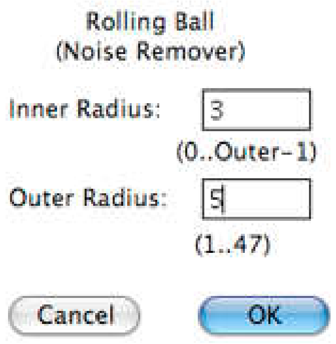
This function ranks the pixel values in a user-adjustable circular neighborhood and keeps the difference between the brightest and darkest values. This may be used for edge or texture finding, depending on the size of the neighborhood and the nature of the image.


Radius (1..7) 3.25000

[✓ Preview]

-> Rolling Ball

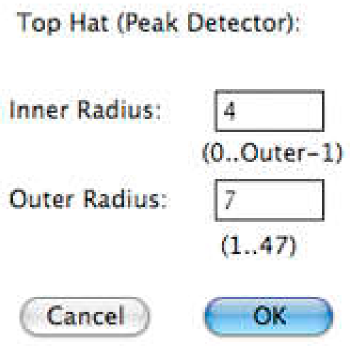
This function uses the logic of the top hat filter (below) but replaces any pixel whose value falls outside the limits set by the brightest and darkest values in the surrounding neighborhood, with the mean value from that outer region. This can be an effective way to remove localized ("shot") noise, dust, etc. The user can adjust the radii for the inside and outside of the surrounding annular neighborhood.


Rolling Ball (Noise Remover)

Inner Radius: 3 (0..Outer-1)

Outer Radius: 5 (1..47)

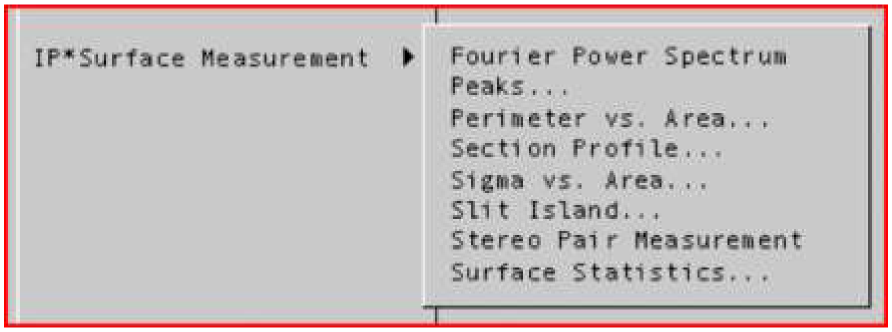
-> Top Hat


Top Hat (Peak Detector):

Inner Radius: 4 (0..Outer-1)

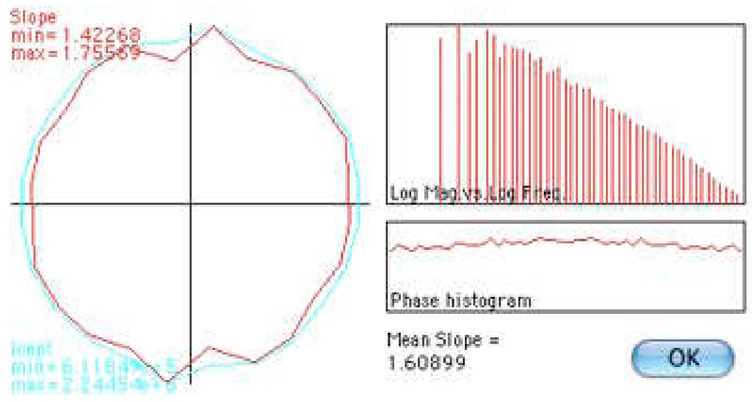
Outer Radius: 7 (1..47)

This function is used to locate bright or dark features of known size. The pixel brightness values are ranked in two neighborhoods, an inner circular region and a surrounding annulus (both user-adjustable in size). If the maximum in the inside is less than that in the outer surround, and the minimum in the inside is greater than that in the outside, the pixel is set to 128 (medium grey). Otherwise the difference between the inner and outer maxima or minima are kept. The result is an image in which local bright and dark spots are kept and can be thresholded. This routine is particularly useful for locating spikes in Fourier power spectra.



This group of functions is used to measure range or elevation images in which grey scale values represent elevation (such images are produced by interference microscopes, scanned probe microscopes, etc.). In most cases these will be 16 bit images. Several of the routines measure fractal dimension of surfaces using different methods (which are not numerically the same). Surface fractal dimensions have values between 2.0 and 2.999… There are also tools to measure conventional surface roughness parameters, either for range (surface) images or cross-sections. These functions are not included in the Image Processing Tool Kit.

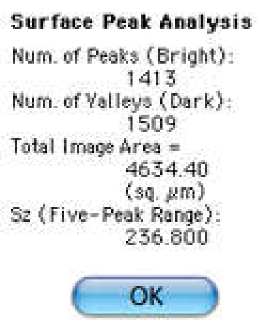
-> Fourier Power Spectrum



This function is used after the IP•Fourier->FFT (Forward) function has been used to obtain the Fourier transform values. The plug-in analyzes the complex data to show several plots that can be used to characterize surface roughness. The plot of log(amplitude) vs. log(frequency) is averaged over all directions; a linear plot indicates a fractal surface, provided that the phase information is random (as shown in the phase plot). The slope and intercept of the log(amplitude) vs. log(frequency) plot are shown as a function of direction to reveal anisotropy in the surface.

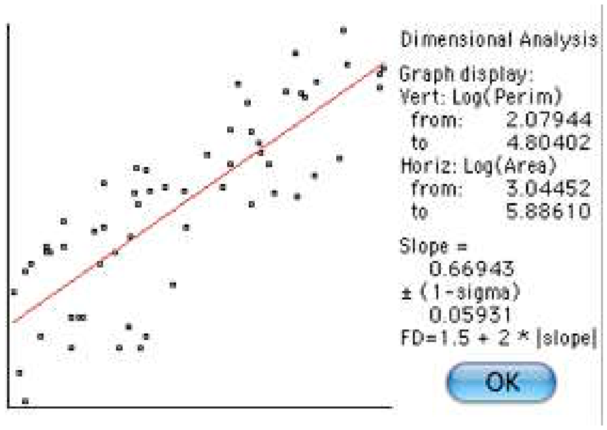
-> Peaks

This function reports the number of distinct peaks and valleys on the surface, and the mean height difference between the five highest peaks and five lowest valleys. The latter value is given in terms of the pixel values (0..255) and should be converted to real units with the elevation calibration information for the surface.


Surface Peak Analysis
Num. of Peaks (Bright): 1413
Num. of Valleys (Dark): 1509
Total Image Area = 4634.40 (sq. µm)
Sz (Five-Peak Range): 236.800

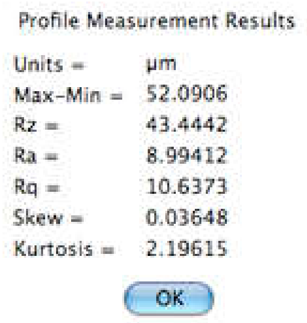
-> Perimeter vs. Area

This function applies one of several methods that measure the fractal dimension of surfaces. The image is thresholded at several heights and the perimeter and area of each feature is measured. A log-log plot of the perimeter vs. area of the features is constructed and the least-squares slope is calculated and reported. This is related to the fractal dimension of the surface as Fractal Dimension = 1.5 + 2 * |slope|.



```
Dimensional Analysis
Graph display:
Vert: Log(Perim)
 from:     2.07944
 to        4.80402
Horiz: Log(Area)
 from:     3.04452
 to        5.88610
Slope =
   0.66943
± (1-sigma)
   0.05931
FD=1.5 + 2 * |slope|
```

-> Section Profile



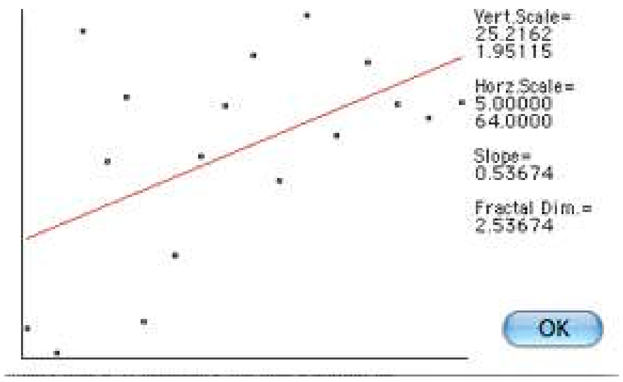
Profile Measurement Results

Units =	µm
Max-Min =	52.0906
Rz =	43.4442
Ra =	8.99412
Rq =	10.6373
Skew =	0.03648
Kurtosis =	2.19615

This function is used with a specific type of image - a vertical cross section through a surface or coating oriented so that the elevation profile is presented as a binary image. If necessary, the image should be rotated so that the profile is oriented in the horizontal direction. The plug-in measures the elevation profile from this image and reports a number of industry-standard surface roughness measurements. Values are given using the spatial calibration in effect.

-> Sigma vs. Area

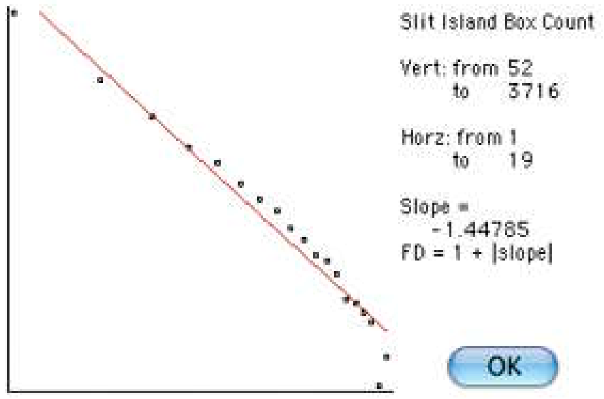
This function applies one of several methods that measure the fractal dimension of surfaces. Randomly located areas of different sizes are measured to determine the variance of the elevation values. A log-log plot of the standard deviation vs. area is constructed and the least-squares slope is calculated and reported. This is related to the fractal dimension of the surface as Fractal Dimension = 2 + slope.



```
Vert.Scale=
25.2162
1.95115
Horz.Scale=
5.00000
64.0000
Slope=
0.53674
Fractal Dim.=
2.53674
```

-> Slit Island

This function applies one of several methods that measure the fractal dimension of surfaces. The image is thresholded and the box-counting or Kolmogorov method is used to determine the fractal dimension of the boundaries produced. A log-log plot of the number of squares occupied by the thresholded islands vs. the size of the grid is constructed and the least-squares slope is calculated and reported. This is related to the fractal dimension of the surface as Fractal Dimension = 1 + |slope|.


Slit Island Box Count

Vert: from 52
to 3716

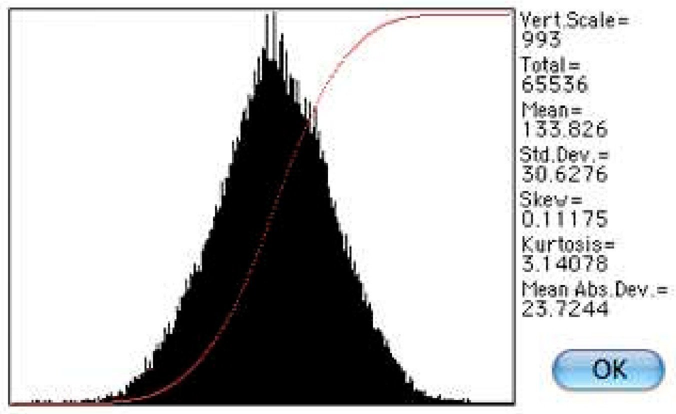
Horz: from 1
to 19

Slope =
-1.44785
FD = 1 + |slope|

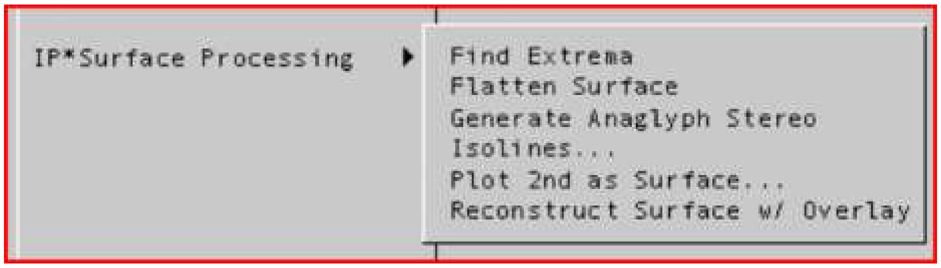
-> Stereo Pair Measurement

This function is used to measure the elevation of points as revealed by their parallax or displacement in stereo pair images. Two images, aligned and at the same magnification, but viewing the same surface from slightly different angles are required. The images must be rotated so that the viewing angle is horizontal (corresponding to the usual way that stereo pair images are viewed). One of the two images should be placed in the second image memory and then this function applied to the other image. Cross-correlation is used to locate matching points in the two images, and the horizontal shift (which is linearly proportional to elevation) is represented by the grey scale value assigned to the pixel. This technique usually produces a noisy result in which some points are not matched or are incorrectly matched, and variations in lighting can exacerbate these problems. A median filter is typically applied afterwards with a large enough neighborhood to erase the outlying values. Converting the grey scale values to absolute elevation measurements can be done by knowing the tilt angle between the images and constructing a "density" calibration curve for the grey scale.

-> Surface Statistics


Vert.Scale=
993
Total=
65536
Mean=
133.826
Std.Dev.=
30.6276
Skew=
0.11175
Kurtosis=
3.14078
Mean Abs.Dev.=
23.7244

This function is used to measure a range image of a surface. It uses the histogram of the image (which is the Abbott-Firestone curve since the values are elevations) to report the statistical values for the surface (industry standard surface roughness parameters). The values are given in terms of the pixel values (0..255) and should be converted to real units with the elevation calibration information for the surface.



Range images can be processed effectively with several of the general image processing functions. A median filter is useful for eliminating dropout points, and a Gaussian filter (or Difference of Gaussians, or Bandpass filter in Fourier space) can be used to perform the typical filtering procedures to separate form, waviness and roughness. The functions grouped here are those that are typically used only for range images. Except for the two surface plotting routines, these functions are not included in the Image Processing Tool Kit.

-> Find Extrema

This function locates and grey scale encodes the ten highest peaks and ten lowest valleys in the image. Differences in the shapes of peaks and valleys indicate asymmetry in the range data and are related to the physical effects that formed the surface. Spatial clustering of peaks and valleys is also diagnostic. The peaks (ranked light grey values) and valleys (ranked dark grey values) are shown with the rest of the area set to medium grey. It is often useful to perform this operation on a duplicate layer so that the corresponding surface regions can be viewed or selected.

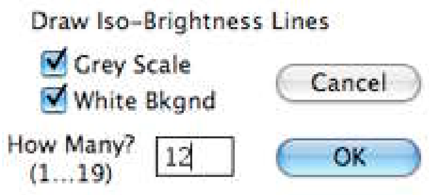
-> Flatten Surface

This function fits a general purpose second degree polynomial to the elevation data and subtracts it from the values. This removes slope and curvature from the data so that correct roughness information can be obtained.

-> Generate Anaglyph Stereo

It is often useful to view surfaces in stereo. This function generates an anaglyph stereo image (to be viewed with colored glasses, red for left eye, green, cyan or blue for right eye) from the elevation data. It is necessary for the image to first be converted to RGB Color before the function is selected.

-> Isolines


Draw Iso-Brightness Lines

✓ Grey Scale

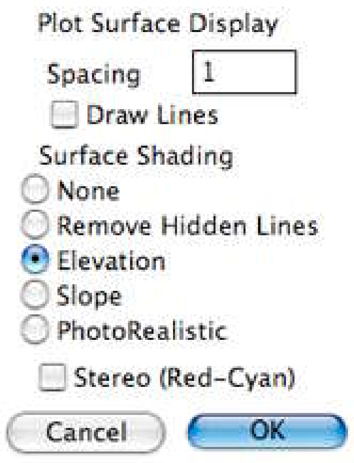
✓ White Bkgnd

How Many? (1...19) 12

Surface elevation is often represented by topographic maps using iso-elevation contour lines. This function converts the range image to a contour map. The lines may be drawn in black on the current image, or on a white background, or can be drawn with their elevation (grey scale) value on a white background. The user can select the number of lines, which are uniformly spaced between maximum and minimum values. Note that the IP•Threshold->Contour Lines routine can also be used with a range image to draw an iso-elevation line at a selected height value.

-> Plot 2nd as Surface

Displaying surface elevation data as a perspective-corrected 3D rendered plot can be an important aid to visualization. This plugin provides several modes for generating the display. First, the range image should be placed in the second image memory. Then a new window of arbitrary size should be created to hold the rendering, and the plug-in selected. The user can choose to have a grid of lines drawn on the surface in the current pencolor, or just the surface itself, and the surface can be shaded according to elevation, local slope, or the product of the two (for photo-realistic rendering of the brightness). If the new window is created in RGB Color mode, the user is given a choice of rendering a stereo pair rather than using color for elevation. In the stereo pair mode, an anaglyph image is created in which two views of the rendered surface are produced, to be viewed with colored glasses.


Plot Surface Display

Spacing 1

☐ Draw Lines

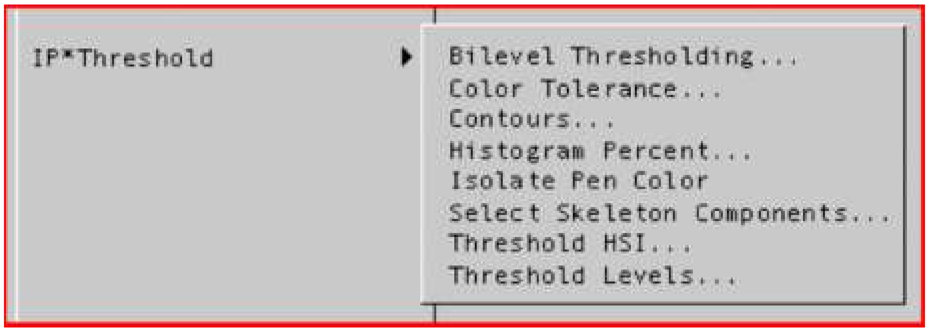
Surface Shading

○ None
○ Remove Hidden Lines
● Elevation
○ Slope
○ PhotoRealistic

☐ Stereo (Red-Cyan)

-> Reconstruct Surface with Overlay

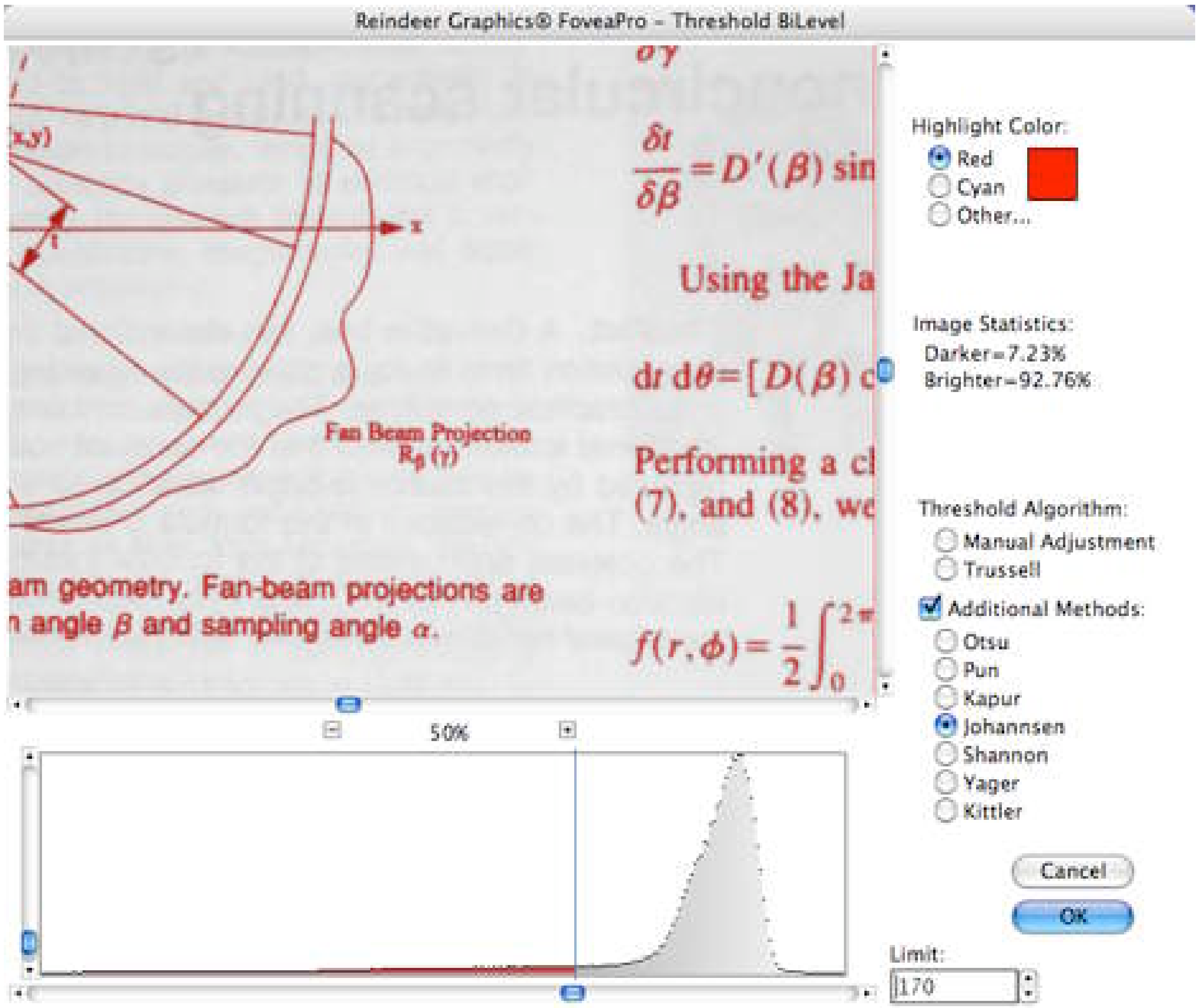
This function uses the same modeling of a perspective-corrected 3D view of the surface as the function above, but rather than rendering the image according to local elevation or slope, it applies the grey scale values from the current image. This is typically used to apply shading such as Phong-rendering, or local detail enhancement produced by image processing of the surface, to the 3D surface model. The procedure is to place the range image containing the elevation data into the second image memory, select the processed image (which must have the same dimensions as the range image), and then apply the function.



Thresholding is the process by which a grey scale or color image is reduced to a binary image, usually black for features and white for background, that delineates the structures or features of interest for measurement. These functions provide different manual and automatic tools for selecting the colors and grey scale values that correspond to those structures. It is expected that prior processing will be applied to correct problems such as shading, and convert edge steps or texture to grey scale, so that thresholding can be performed. Additional morphological processing after thresholding may also be needed.

In addition to the routines presented in this category, the IP•Surface Processing -> Isolines and the IP•Multichannel -> Channel Thresholding routines also provide thresholding capabilties.

-> Bilevel Thresholding


Reindeer Graphics® FoveaPro - Threshold BiLevel

Highlight Color: [✓ Red] Cyan Other...

Image Statistics:
Darker=7.23%
Brighter=92.76%

Threshold Algorithm: Manual Adjustment Trussell

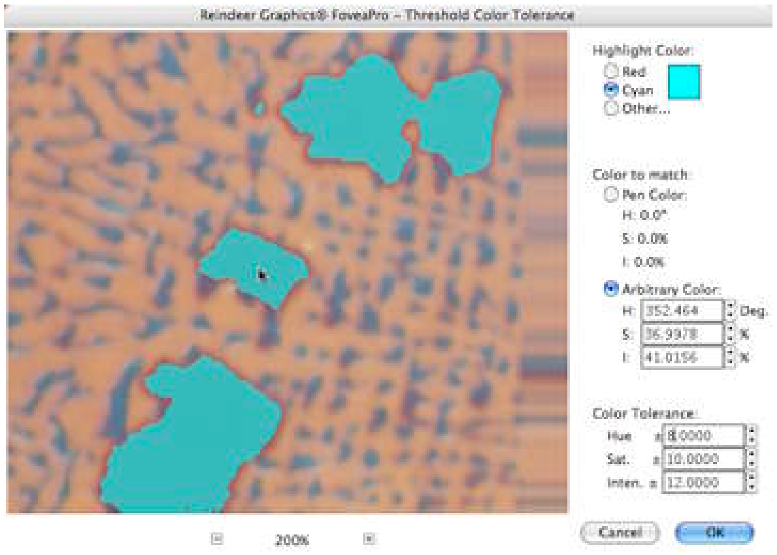
[✓ Additional Methods:] Otsu Pun Kapur [✓ Johannsen] Shannon Yager Kittler

Limit: 170

Similar to the built-in Photoshop Image->Adjustments->Threshold routine, this function divides the image into bright and dark pixels. The user can adjust the position of the threshold by reference to the histogram, and the area fractions of the image thus selected are shown. Originally developed for images of print on paper, the “auto” button applies a statistical test (t-test) to determine an optimum position for the threshold. The assumption is that the image consists of just two classes of pixels that are to be distinguished based on brightness. Optionally, the user can also access a selection of alternative automatic thresholding algorithms (named for the authors of the original publications). These differ in the assumptions they make about the distributions of brightness values and the statistical tests that are applied; their success varies with different applications.

-> Color Tolerance

This function is used primarily with RGB color images. Either the pen color, or a color selected from the image can be selected as the target color. The dialog reports this color (as hue, saturation and intensity) and allows setting tolerances separately on each of those parameters to define the structure. This function is an efficient way to implement HSI thresholding once the nature of the colors in the image is well understood, for instance by using the Threshold HSI function. It also provides for interactive selection of features based on their colors.



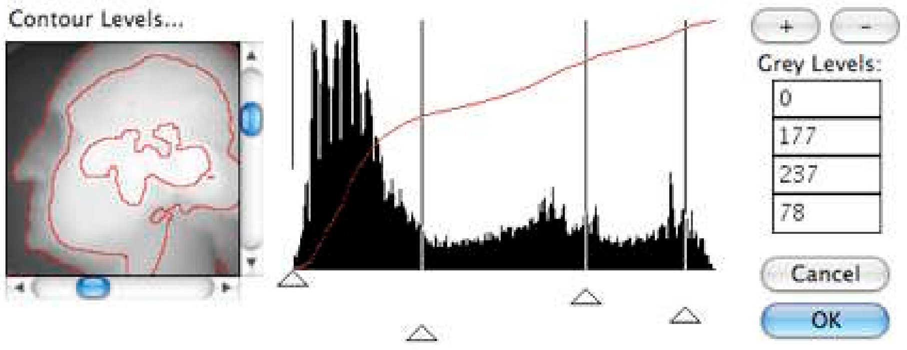
Highlight Color: ( ) Red [✓ Cyan] ( ) Other...

Color to match:
( ) Pen Color: H: 0.0° S: 0.0% I: 0.0%
[✓ Arbitrary Color:] H: 352.464 Deg. S: 36.9978 % I: 41.0156 %

Color Tolerance:
Hue ± 8.0000
Sat. ± 10.0000
Inten. ± 12.0000

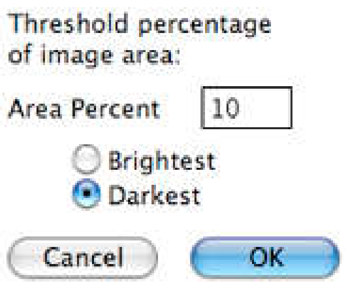
-> Contours

This function draws from 1 to 4 isobrightness lines on the image at brightness values selected by the user by positioning markers on the histogram. Isobrightness lines are particularly useful because are continuous and frequently correspond to boundaries around structures or features.



Grey Levels:
0
177
237
78

-> Histogram Percent



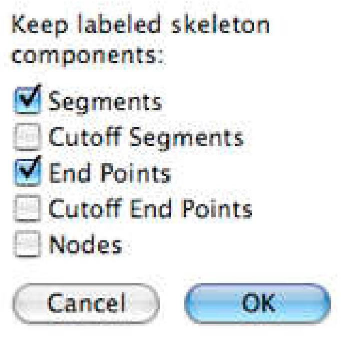
Threshold percentage of image area:
Area Percent 10
( ) Brightest [✓ Darkest]

This function allows selecting pixels that represent the brightest or darkest fixed percentage of the image. It is most commonly used in automatic actions on FFT power spectra, or on Top hat filtered images (particularly after the Adjust->Limits function has been used) to select a consistent fraction of the pixels in the frame.

-> Isolate Pen Color

This function is typically used after drawing lines or marks on an image which are to be counted or measured. It converts pixels in the current pen color to black and erases all others, so that the lines are isolated for measurement. It is equivalent to using the Color Tolerance function with the tolerances set to zero. In many cases, a good procedure is to duplicate the original image in a layer before applying this function, so that the resulting lines and marks can be seen as an overlay on the original image.

-> Select Skeleton Components

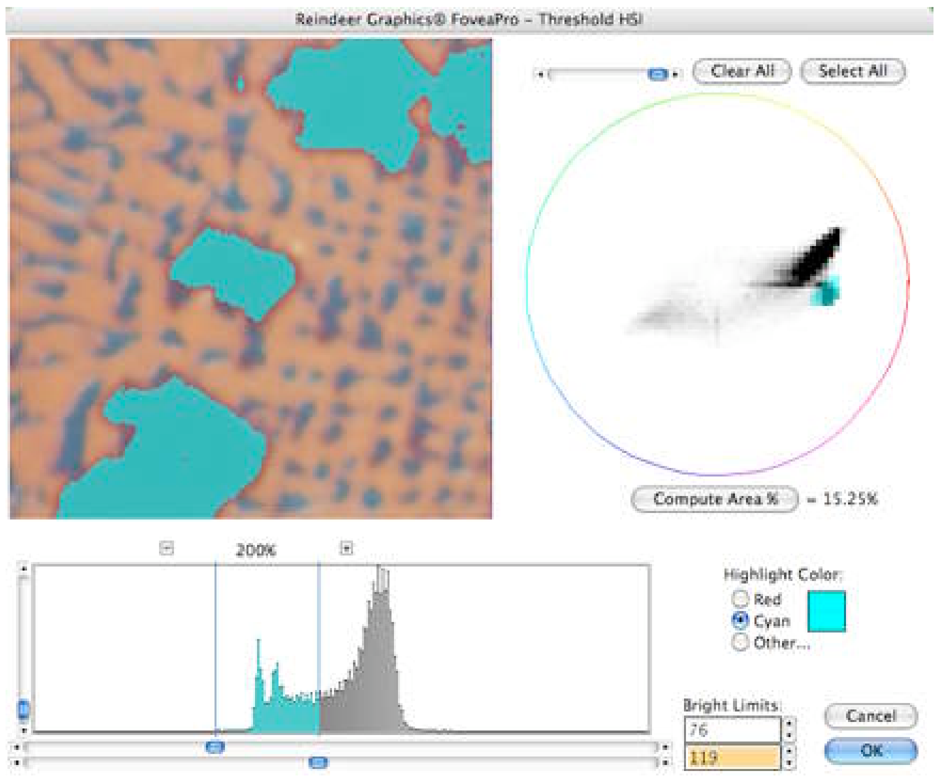


Keep labeled skeleton components:

- [✓] Segments
- [ ] Cutoff Segments
- [✓] End Points
- [ ] Cutoff End Points
- [ ] Nodes

This function should be used after the IP•Morphology->Skeletonize or ->Skeletonize and Cutoff plug-in has been applied to an image. Those procedures code the pixels in the skeleton with unique grey scale values that distinguish the end points, branch points, and (if Cutoff was used) the cut-off branches and end points from the main body of the skeleton. This function makes it easy to select which of those components are to be kept and which are to be removed, so that counting or measurement, or use of the features as markers can be performed.

-> Threshold HSI



Reindeer Graphics® FoveaPro – Threshold HSI

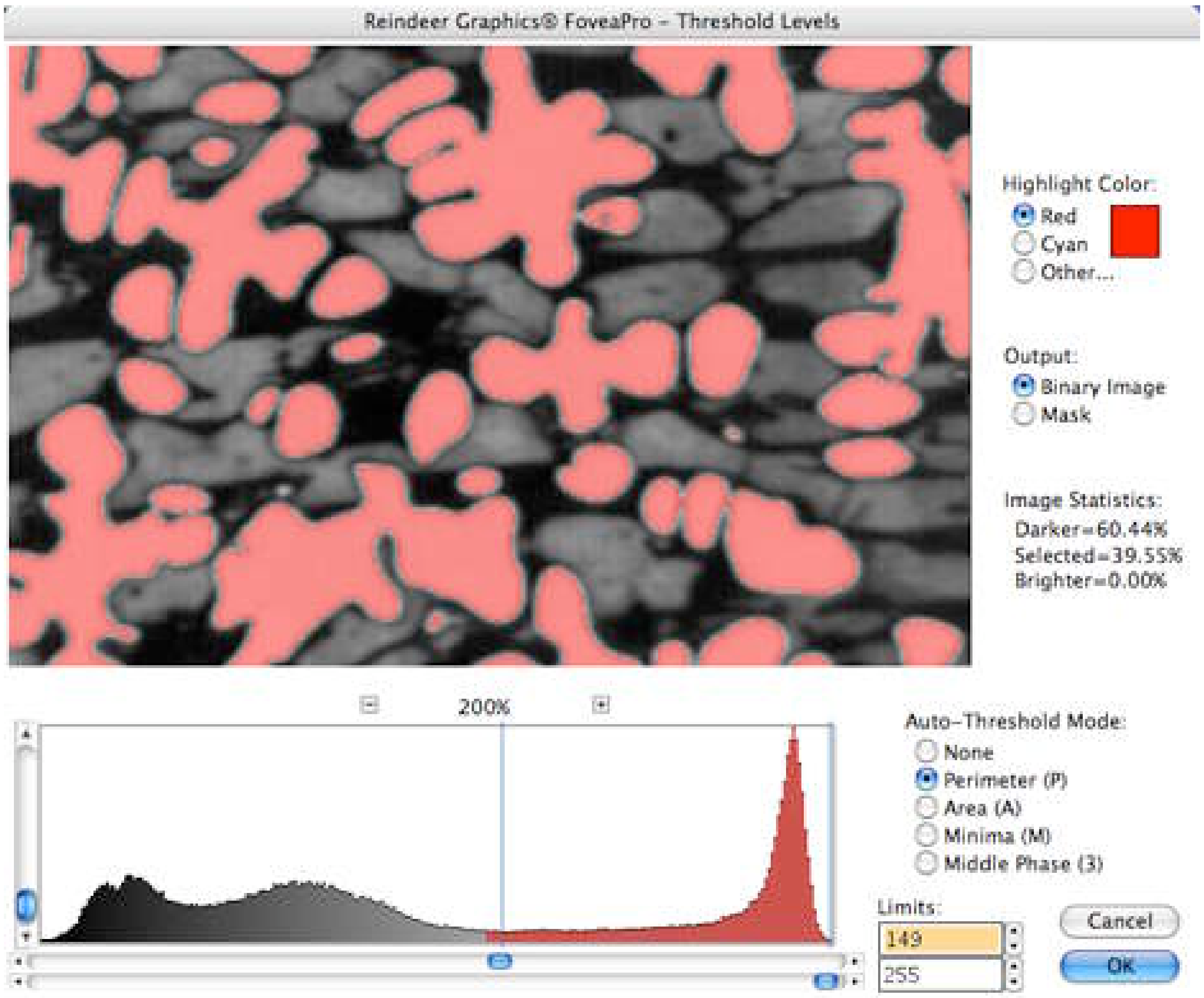
Compute Area % = 15.25%

Highlight Color: Red [✓ Cyan] Other...

Bright Limits: 76, 119

Color images are generally easiest to threshold when represented in hue-saturation-intensity space rather than by their RGB values. This plugin generates a histogram of the HSI values in which the hue and saturation are shown on a circular color wheel, and the (adjustable) darkness of each cell in the wheel represents the number of pixels, along with a traditional histogram for brightness. It is possible to select arbitrary, even disconnected regions of the H-S wheel by marking the corresponding cells, as well as setting the usual limits on the brightness values. As an aid to understanding the color representation, clicking on any point in the image preview will mark the corresponding HSI points on the histograms. Also, the preview shows the thresholded pixels in a user-selected color.

-> Threshold Levels


Reindeer Graphics® FoveaPro – Threshold Levels

Highlight Color: [✓ Red] Cyan Other...

Output: [✓ Binary Image] Mask

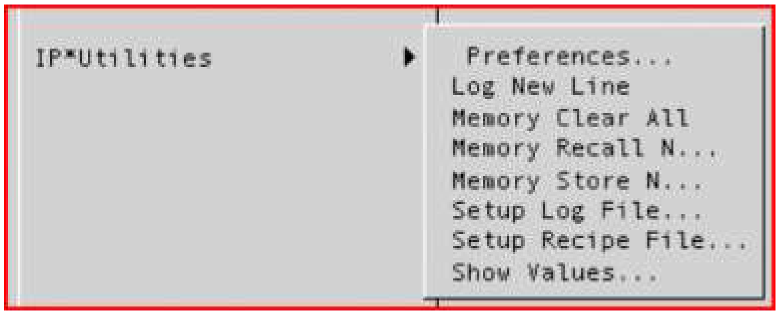
Image Statistics:
Darker=60.44%
Selected=39.55%
Brighter=0.00%

200%

Auto-Threshold Mode: None [✓ Perimeter (P)] Area (A) Minima (M) Middle Phase (3)

Limits: 149 255

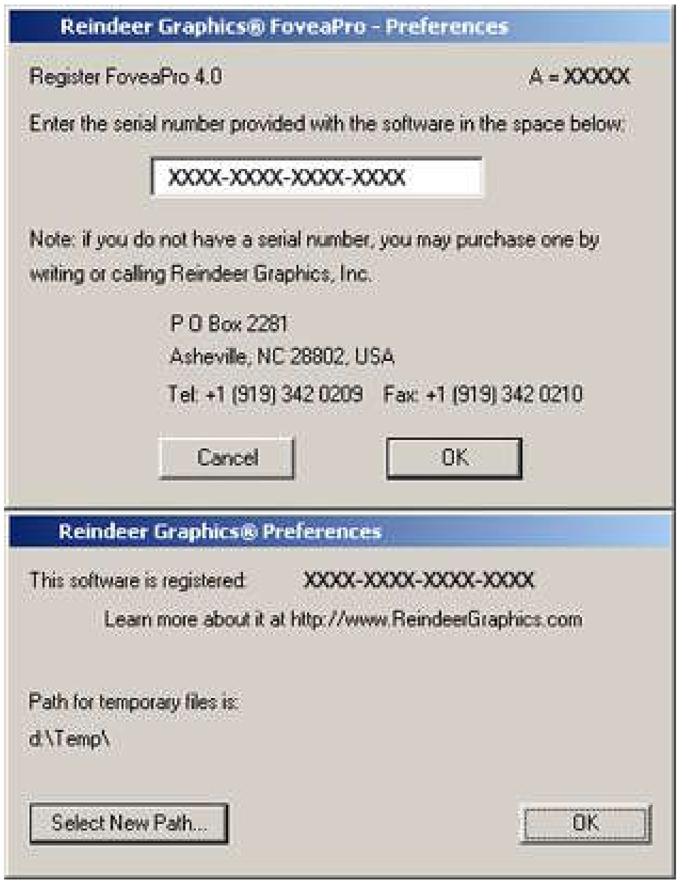
This function shows the brightness histogram of the image with two user-adjustable sliders. These mark the upper and lower limits of the brightness slice that is selected, and the dialog displays the area fractions each slice represents. The preview can be selected to show the selected pixels in color, or to erase the unselected pixels. There are several buttons that can be used to modify the user selection. The Area, Perimeter and Minimum buttons will move the threshold settings by up to 16 grey levels from the user-selected position (unless the current position is exactly at one of the ends, where it is left unchanged) according to the selected criteria. The Perimeter button chooses the point at which the perimeters of the features are smoothest. The Area button chooses the point at which the features have the most consistent areas. The Minimum button finds the point at which the histogram is lowest. The 3-Phase button applies an analysis of variance criterion to select the three bands of brightness that have the highest statistical probability of representing three different populations. All of these methods presume that the user has independent knowledge of the specimen on which to base a selection of the proper technique.



The utility functions are mostly routines pertaining to automation and used with Photoshop actions. Except for the Preferences function they are not included in the Image Processing Tool Kit.

-> Preferences

This plugin is used to enter the serial number for the software. If no serial number is entered, the plug-ins will function for 20 days after installation. Serial numbers can be obtained from Reindeer Graphics Inc. (The A= number, which serves as a machine identification, is needed to obtain a valid number.)


Reindeer Graphics® FoveaPro - Preferences

Register FoveaPro 4.0 A = XXXXX

Enter the serial number provided with the software in the space below:

XXXX-XXXX-XXXX-XXXX

Note: if you do not have a serial number, you may purchase one by writing or calling Reindeer Graphics, Inc.

P O Box 2281
Asheville, NC 28802, USA
Tel: +1 (919) 342 0209 Fax: +1 (919) 342 0210

Reindeer Graphics® Preferences

This software is registered: XXXX-XXXX-XXXX-XXXX

Learn more about it at http://www.ReindeerGraphics.com

Path for temporary files is:
d:\Temp\

This function is also used on the PC (but not on the Mac) to specify the location for scratch files used by the plug-ins. The default location for temporary files, preferences, etc., is the C:\ drive on the PC, but can be redirected elsewhere if necessary (e.g., to allow the C:\ drive to be protected from write access or to use a drive with more available space than the boot volume). Select a location and enter the name for any file there (the actual name is not used, just the path).

Access to a fast, local drive is necessary. Note that this does not affect the location used by Photoshop for its scratch files (selected with Preferences -> Plug-ins and Scratch Disks).

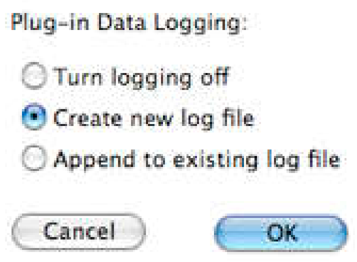
-> Log New Line

When logging is turned on (see Setup Log File, below), this function writes the current filename and the current date and time to the file, and then writes a carriage return. This is typically used at the end of an action so that when it is applied to a batch processing of a folder full of images, the data file will consist of one line of data for each image, arranged in columns and suitable for spreadsheet analysis.

-> Memory Clear All
-> Memory Recall N
-> Memory Store N

For automation purposes it is sometimes useful to have access to temporary storage other than that which can be accomplished by duplicating images or image layers. Running actions in batch mode can open images only from one folder, and each open command accesses the next image in the list. Selecting between multiple open images or layers requires knowing their exact current order. The use of separate, explicitly addressed image memories allows keeping grids available, preserving intermediate results, and comparing one image to prior images (e.g., for serial section analysis such as the disector). Fovea Pro provides ten explicit image memories, numbered 1 through 10, that can be used for these purposes. The contents of the memories are preserved when exiting Photoshop as they are stored to disk. The menu functions clear all memories, store an image in any one of the memories (which need not be filled in any particular order), and recall an image from any memory (a dialog shows the image contents for the memories in current use). If the stored image is a different size or mode (grayscale vs RGB, 8 or 16 bits) than the current image, the data are converted to the current mode and the common rectangle at the upper left corner is used. Note that these image memories are entirely separate from and have no relationship to the "second image" storage used for two-image operations such as mathematics functions.

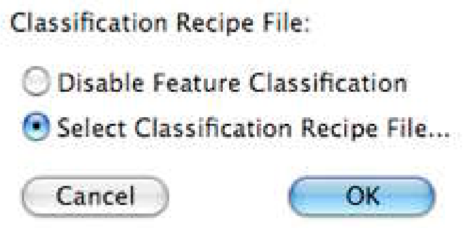
-> Setup Log File


Plug-in Data Logging:
Turn logging off
[✓ Create new log file]
Append to existing log file

This function allows the selection of a log file (or turning off the logging function). When turned on, logging saves the various measurement results, with labels, to a tab-delimited text file that can later be opened in a spreadsheet or data analysis program. This is typically used with batch processing of images using an action, but can also be useful for accumulating data from various measurements without having to write down the information displayed in measurement dialogs. Logging also saves a folder containing an html file (index.html) that can be read with your web browser, and includes the graphics from those measurement routines that report graphics in dialogs (which are also saved as individual *.tif files).

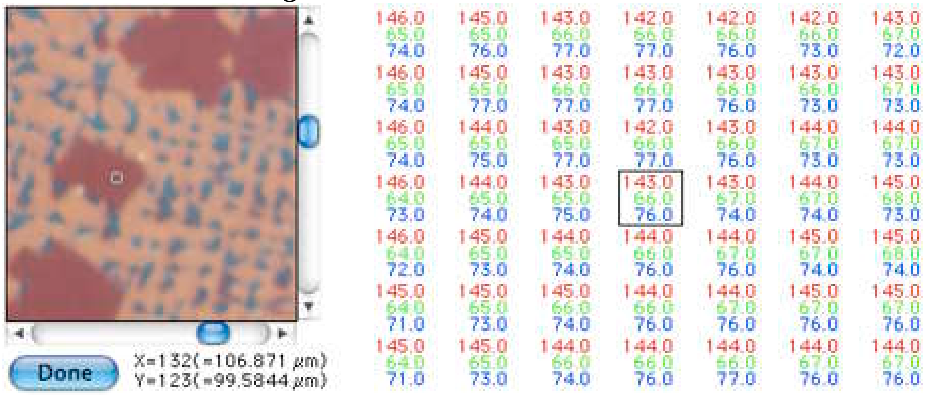
-> Setup Recipe File

This function allows the selection of a recipe file. These files contain the parameter names and numeric limits that define multiple classes for feature identification. The files are tab-delimited text files that can be created with a spreadsheet or word processor.


Classification Recipe File:
Disable Feature Classification
[✓ Select Classification Recipe File...]

-> Show Values

This function is like the Photoshop info display and shows the grey scale or RGB values for pixels selected in the preview window, as well as their coordinates (in the current calibrated units) from the upper left corner of the image. The built-in function shows only integer 0..255 values, which cannot represent the full precision of 16 bit per channel images. This plug-in corrects that limitation, and also makes it possible to record the values to the log file.



```
146.0  145.0  143.0  142.0  142.0  142.0  143.0
 65.0   65.0   66.0   66.0   66.0   66.0   67.0
 74.0   76.0   77.0   77.0   76.0   73.0   72.0
146.0  145.0  143.0  143.0  143.0  143.0  143.0
 65.0   65.0   66.0   66.0   66.0   66.0   67.0
 74.0   77.0   77.0   77.0   76.0   73.0   73.0
146.0  144.0  143.0  142.0  143.0  144.0  144.0
 65.0   65.0   65.0   66.0   66.0   67.0   67.0
 74.0   75.0   77.0   77.0   76.0   73.0   73.0
146.0  144.0  143.0  143.0  143.0  144.0  145.0
 64.0   65.0   65.0   66.0   67.0   67.0   68.0
 73.0   74.0   75.0   76.0   74.0   74.0   73.0
146.0  145.0  144.0  144.0  144.0  145.0  145.0
 64.0   65.0   65.0   66.0   67.0   67.0   68.0
 72.0   73.0   74.0   76.0   76.0   74.0   74.0
145.0  145.0  145.0  144.0  144.0  145.0  145.0
 64.0   65.0   66.0   66.0   67.0   67.0   67.0
 71.0   73.0   74.0   76.0   76.0   76.0   76.0
145.0  145.0  144.0  144.0  144.0  144.0  144.0
 64.0   65.0   66.0   66.0   66.0   67.0   67.0
 71.0   73.0   74.0   76.0   77.0   76.0   76.0
```