IMG* Image Processing Toolset Manual
Version 1.1 (Nov 1994)

Copyright (C) 1994 Simon A. J. Winder
School of Mathematical Sciences
The University, Bath, BA2 7AY, England
E-mail: sajw@maths.bath.ac.uk

November 23, 1994

Introduction

This document contains a description of all the image processing tools in the "img*" series. This set was intended to complement the PbmPlus tool-set and the XV and XPAINT public domain programs. Many of the tools use a floating point image format and there are tools to convert between this and the popular PPM, PGM and PBM types as necessary.

These tools are all used from the UNIX command line and act as filters through which image data is piped. In this way, a sequence of processing operations can be set up on one command line using the UNIX pipe facility. Standard input and output redirection is used in most cases to read and write image data to files. Many of the tools also have optional arguments. These are specified in quite a primitive but simple way and argument details can be found in the manual pages for each tool as needed. **All tools include a "-h" option which prints out usage information.** Arguments shown in square brackets in this manual are optional.

A significant feature of this tool-set is the implementation of image sequence handling. The majority of tools are able to apply their operation to one or more images passing through from *stdin* to *stdout*. More than one image can thus be processed on the same command line. This is achieved by using a tool that gathers images from a number of randomly-named or sequentially-named files and sends them out as a sequence. This sequence then passes through all the desired filters. Another tool then separates the individual images and saves them to a set of filenames. Various other sequence-handling tools will change the sequence order, tee-out or interleave-in other image streams etc. Operations like frame-differencing can be carried out using a single command line.

Examples

```
imgPnmToFlt <img.pgm | imgSmooth 2 | imgCanny | imgFltToPgm | xv –
```

This will carry out the first stages of Canny edge detection on the image named *img.pgm* and display the result using XV.

```
imgPnmToFlt <img.ppm | imgForwardFFT | imgFFTHigh | imgInverseFFT
| imgFltToPgm >out.pgm
```

This will carry out high-pass filtering in the Fourier domain and save the resultant in *out.pgm*.

```
imgPpmToFlt3 <img.ppm | imgSmooth 2.5 | imgFlt3ToPpm >out.ppm
```

This command sequence smooths a colour image.

```
imgRankFilter 5 <in.flt >out.flt
```

This will carry out median filtering on the floating point image *in.flt* and save it in *out.flt*.

```
imgCombine fred.flt jim.flt sam.flt | imgSubSample 2
| imgSplit fred_ss.flt jim_ss.flt sam_ss.flt
```

This command sequence loads the three floating point images given, subsamples them by a factor of two and saves them with different names.

```
imgCombineSeq seq.pgm 0 1 9 | imgInsert seq.pgm 1 1 9 1 2 | imgSub
| imgSplitSeq out.pgm 0
```

This command line frame-differences the ten images stored in files *seq000.pgm*, *seq001.pgm* to *seq009.pgm* and places the nine resultant images in files named *out000.pgm* to *out008.pgm*.

List of Tools

imgAbs	Calculate pixel absolute values.
imgAdd	Add images or add constant to image.
imgAnd	Perform logical AND between bit images.
imgAngle	Calculate polar phase information.
imgAscii	Convert image into an ASCII representation.
imgCanny	Perform Canny edge detection.
imgCombine	Combine images from files to form a sequence.
imgCombineSeq	Combine images from sequenced files to form a sequence.
imgConvolve	Convolve float image with a kernel.
imgCpxMul	Multiply two complex images.
imgCpxToFlt2	Convert complex image into floating point images.
imgCrop	Extract a subregion from each input image.
imgDeriv	Compute the one dimensional directional derivative of an image.
imgDiv	Divide images or divide image by a constant.
imgEdgeSynth	Canny two-level edge synthesis operation.
imgFFTBandPass	Fourier Gaussian bandpass filter.
imgFFTDOG	Fourier difference of Gaussian filter.
imgFFTGabor	Fourier directional Gabor filter.
imgFFTHigh	Fourier low cut, high-pass filter.
imgFFTLow	Fourier high cut, low-pass filter.
imgFFTSmooth	Fourier Gaussian smoothing filter.
imgFlip	Reflect image horizontally or vertically.
imgFlt2ToCpx	Convert two floating point images into complex format.
imgFlt3ToPpm	Convert three floating point images into PbmPlus type PPM images.
imgFltToPgm	Convert floating point images into PbmPlus type PGM images.
imgForwardFFT	Calculate the forward fast Fourier transform of an image.
imgFourierPad	Convert a convolution kernel into a form useful for Fourier processing.
imgFoveate	Foveally smooth image with a Gaussian kernel.
imgGrad	Calculate image gradient vector at each pixel.
imgGrating	Generate a sine-wave grating for test purposes.
imgImage	Generate a blank image.

imgImpulse	Generate a single impulse for test purposes.
imgInfo	Show information about the current image.
imgInsert	Insert images from sequenced files into the image stream.
imgInverseFFT	Calculate the inverse fast Fourier transform of an image.
imgLaplacian	Calculate image Laplacian at each pixel.
imgMagnify	Increase pixel size of input images.
imgMagnitude	Calculate the magnitude of a pixel vector.
imgMarrHil	Marr-Hildreth type edge operator.
imgMax	Find maximum pixel values over N images.
imgMul	Multiply images or multiply image by a constant.
imgNeg	Negate image values.
imgNoise	Add Gaussian random noise to a floating point image.
imgNot	Perform logical NOT of a bit image.
imgOr	Perform logical OR between bit images.
imgPnmToFlt	Convert PbmPlus type images into floating point format.
imgPower	Raise image values to a constant power.
imgPpmToColour	Convert PPM image into two orthogonal colour.
imgPpmToFlt3	Convert PPM colour images into three floating point format images.
imgPrint	Print contents of an image sub-region.
imgRankFilter	Perform rank-value filtering on a floating point image.
imgRectify	Half-wave rectify all image values.
imgReorder	Re-arrange floating point image for Fourier transform viewing.
imgRepeat	Produces a repeated sequence of each input image.
imgSetMinMax	Set minimum and maximum value fields of a floating point image.
imgSelect	Pick out a set of images from the stream.
imgSmooth	Smooth image with a Gaussian kernel.
imgSplit	Write images from the input stream to files.
imgSplitSeq	Write images from the input stream to sequenced filenames.
imgStats	Calculate statistical information.
imgSub	Subtract images or subtract constant from image.
imgSubSample	Perform sub-sampling on each image.
imgSurfacePlot	Surface viewing tool for X Windows.

imgTee	Tee selected images from the stream to sequenced filenames.
imgThresh	Threshold image.
imgWrapAngle	Adjust angle data to lie in range $\pm\pi$.
imgXor	Perform logical XOR between bit images.
imgZeros	Zero-crossing detector.

imgAbs

Name

imgAbs—Calculate pixel absolute values.

Syntax

imgAbs

Image Types

stdin	Float format
stdout	Float format

Description

For each image in the input sequence, imgAbs computes the floating point absolute value on a per-pixel basis. This yields a full-wave rectified result.

Treatment of Valid Region

The valid region is not changed by this operator, but is ignored for the purposes of processing. The process is applied to the whole image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are obtained during processing and are propagated with the output image.

See Also

imgRectify, imgAdd, imgMul, imgDiv, imgSub, imgNeg.

imgAdd

Name

imgAdd—Add images or add constant to image.

Syntax

imgAdd [constant]
imgAdd [filename]

Image Types

stdin	Float format
stdout	Float format
file	Float format

Description

This process performs a pixel-wise addition of images. Each input image is processed in turn and addition is performed in one of three ways: Either a constant provided on the command line is added to all pixel values, or the image file named on the command line is added in a pixel-wise manner, or else the next input image is added in a pixel-wise manner. In the case of addition of two images, the image dimensions must be the same for both images.

Arguments

imgAdd with no arguments adds image pairs from *stdin*. If a numeric is specified on the command line, this is interpreted as a constant to add. Otherwise, this argument is interpreted as the filename of an image to load and add.

Treatment of Valid Region

The valid region is not changed by this operator, but is ignored for the purposes of processing. The process is applied to the whole image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are obtained during processing and are propagated with the output image.

See Also

imgAbs, imgMul, imgDiv, imgSub, imgNeg.

imgAnd

Name

imgAnd—Perform logical AND between bit images.

Syntax

imgAnd [filename]

Image Types

stdin	Bit format (pbm)
stdout	Bit format (pbm)
file	Bit format (pbm)

Description

This process performs a pixel-wise logical AND of images. Each input image is processed in turn and the operation is performed in one of two ways: Either the image file named on the command line is ANDed with the input image in a pixel-wise manner, or else the next input image is ANDed in a pixel-wise manner. The image dimensions must be the same for both images.

Arguments

imgAnd with no arguments operates on image pairs from *stdin*. Otherwise, the argument is interpreted as the filename of an image to load.

Treatment of Valid Region

Valid regions are not supported with this image type.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are not supported with this image type.

See Also

imgOr, imgXor, imgNot.

imgAngle

Name

imgAngle—Calculate polar phase information.

Syntax

imgAngle

Image Types

stdin	Complex format, Polar format
stdout	Float format

Description

If imgAngle inputs a complex type image, a rectangular to polar conversion is used to give the polar argument in the range $-\pi$ to π. If the input image is of polar type, this command simply extracts the polar argument and outputs it directly.

Treatment of Valid Region

The input valid region is propagated unchanged by this command.

Treatment of Minimum and Maximum Values

The minimum and maximum values are calculated and set by this command.

See Also

imgMagnitude, imgDeriv, imgGrad, imgLaplacian, imgWrapAngle.

imgAscii

Name

imgAscii—Convert image into an ASCII representation.

Syntax

imgAscii

Image Types

stdin	Any format
stdout	Any format

Description

imgAscii converts the image representation into an ASCII form of the same image type as the input. This ASCII form is suitable for printing to the terminal, loading into editors and mailing. Note that for any reasonable sized floating point image, this conversion yields a huge amount of data. Pixel values are printed as decimal values separated by spaces on lines not exceeding 70 characters in length.

Treatment of Valid Region

The valid region is not changed by this operation.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values (if present) are propagated with the output image.

See Also

imgPrint, imgInfo, imgStats.

imgCanny

Name

imgCanny—Perform Canny Edge Detection.

Syntax

imgCanny

Image Types

stdin	Float format
stdout	Float format

Description

This command implements the edge detection process proposed by J. Canny. The complete process is Gaussian smoothing using imgSmooth followed by this process, followed by two-level thresholding and edge synthesis. A 3×3 kernel is used and the output points are in registration with the kernel centre for any kernel position. The output is a set of single-pixel wide edges which have as their value the magnitude of the gradient at that location. These edges are surrounded by a sea of value 0.0.

Treatment of Valid Region

The output valid region is set to the valid region defined for the first image in each group with an adjustment for the kernel size.

Treatment of Minimum and Maximum Values

The minimum and maximum values are calculated during processing.

See Also

imgMarrHil, imgZeros, imgEdgeSynth.

imgCombine

Name

imgCombine—Combine images from files to form a sequence.

Syntax

imgCombine file1 [file2 [...]]

Image Types

stdin	None
stdout	Any format
file	Any format

Description

imgCombine loads the image files specified on its command line in the order given, performs validation and outputs them in succession. This is functionally the same as UNIX "cat" except that an error status will be returned if any image is not a recognised type.

Arguments

Command line arguments specify the files to be combined.

Treatment of Valid Region

The valid region information is propagated unaltered by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCombineSeq, imgInsert, imgRepeat, imgSelect, imgSplit, imgSplitSeq, imgTee.

imgCombineSeq

Name

imgCombineSeq—Combine images from sequenced files to form a sequence.

Syntax

imgCombineSeq basename [start [incr [count]]]

Image Types

stdin	None
stdout	Any format
file	Any format

Description

imgCombineSeq loads the image files specified by the command line parameters and outputs them in sequence to *stdout*. The image files form a sequence with three digits in the filename indicating the sequence number. Files are loaded in ascending order starting from the file with sequence number *start* and incrementing by *incr* each time. At most *count* files are loaded.

Arguments

The filename argument specifies the basename of the file sequence. If a dot extension is specified, then the three digit sequence number will be inserted after the basename and before this extension. With no further command options, the default action is to load the file exactly as given, without a sequence number. If the *start* parameter is given, this sets the sequence number of the first file to be loaded. With no further options, imgCombineSeq loads files consecutively from this number until no further file is found. If an increment is specified then this is used to generate further sequence numbers. The parameter *incr* can be negative, but note that a file sequence number cannot be negative. The final parameter specifies the number of files to load.

Treatment of Valid Region

The valid region information is propagated unaltered by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCombine, imgInsert, imgRepeat, imgSelect, imgSplit, imgSplitSeq, imgTee.

imgConvolve

Name

imgConvolve—Convolve float image with a kernel.

Syntax

imgConvolve filename

Image Types

stdin	Float format
stdout	Float format
file	Double format

Description

imgConvolove performs space domain convolution between the input image and the convolution kernel from the file whose name is supplied on the command line. The kernel file can be created using an editor and has the following format:

```
ILibDouble

kernel_width kernel_height

0 0 0 0 0 0

kernel_data
```

The kernel width and height fields are ASCII integers and the data field is a white-space separated list of width×height numbers which can be floating point.

Arguments

The filename argument specifies the kernel file. Each incoming image will be convolved with this.

Treatment of Valid Region

The valid region information from each input image is reduced in size appropriately.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are calculated for the valid output region.

See Also

imgSmooth, imgForwardFFT, imgInverseFFT.

imgCpxMul

Name

imgCpxMul—Multiply two complex images.

Syntax

imgCpxMul [filename]

Image Types

stdin	Complex format
stdout	Complex format
file	Complex format

Description

imgCpxMul multiplies two equally sized complex images together to yield a resultant complex image. The chief use for this process is in convolution by frequency domain multiplication in conjunction with the Fourier transform tools.

Arguments

The filename argument specifies the second image in the multiplication. Each incoming image will be multiplied by this. If no filename argument is supplied then pairs of input images will be multiplied.

Treatment of Valid Region

The valid region information from the first image in each pair is propagated to the output and is unaltered by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are not set by this function.

See Also

imgForwardFFT, imgInverseFFT.

imgCpxToFlt2

Name

imgCpxToFlt2—Convert complex image into floating point images.

Syntax

imgCpxToFlt2

Image Types

stdin	Complex format
stdout	Float format

Description

imgCpxToFlt2 converts one complex format image into two successive floating point images. The first output image is the real part and the second is the imaginary part.

Treatment of Valid Region

The valid region from the complex image is propagated to both the output images. The valid region information is ignored for the purposes of copying the data.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values in the input image are not used. New minimum and maximum values are computed separately for the real and imaginary output images during the copying process.

See Also

imgPnmToFlt, imgFltToPgm, imgPpmToColour, imgPpmToFlt3, imgFlt3ToPpm, imgFlt2ToCpx.

imgCrop

Name

imgCrop—Extract a subregion from each input image.

Syntax

imgCrop [x y width height]

Image Types

stdin	Any format
stdout	Any format

Description

imgCrop extracts a rectangular region from each image and outputs this as a new image. The command line specifies the region to extract. If the arguments are missing, imgCrop uses the input image's valid region information to set the cropping region.

Arguments

Command line arguments specify the top left coordinates and width and height of the region to extract.

Treatment of Valid Region

The output valid region is set to the whole of the output image to avoid non-intersection problems.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function and are not recomputed. This allows image display contrast to be independent of the cropped region when using imgFltToPgm.

See Also

imgMagnify, imgReorder, imgSubSample.

imgDeriv

Name

imgDeriv—Compute the one dimensional directional derivative of an image.

Syntax

imgDeriv [angle]
imgDeriv [filename]

Image Types

stdin	Float format
stdout	Float format
file	Float format

Description

imgDeriv calculates the one dimensional first derivative of the image intensity values in the direction specified. The direction can either be a constant for the whole image, or else be defined at each pixel. This operator uses a 2×2 kernel and calculates two diagonal orthogonal differences, scaled appropriately. From these components, the required directional derivative is calculated. The result obtained by applying the kernel is placed at the position overlaid by the top left kernel pixel for each position in the output image.

Arguments

Without an argument, the default action is to accept pairs of input images. The first is the image to be processed and the second contains orientation data for each pixel. The images must be the same size. If a numeric argument is given, this is interpreted as a constant angle measured in degrees. The image will be differentiated in this direction and no orientation file will be read. The argument can also be a filename, in which case the orientation image will be read from this source.

Treatment of Valid Region

Processing only takes place within the input valid region. The valid region of any angle image is ignored. The Output valid region is one pixel smaller in width and height than the input region.

Treatment of Minimum and Maximum Values

Minimum and maximum values are calculated from the results of this process applied within the valid input region.

See Also

imgAngle, imgMagnitude, imgGrad, imgLaplacian.

imgDiv

Name

imgDiv—Divide images or divide image by a constant.

Syntax

imgDiv [constant]
imgDiv [filename]

Image Types

stdin	Float format
stdout	Float format
file	Float format

Description

This process performs a pixel-wise division of images. Each input image is processed in turn and division is performed in one of three ways: Either all pixel values are divided by a (non-zero) constant provided on the command line, or the image file named on the command line is treated as the denominator in a pixel-wise manner, or else the next input image is treated as the denominator. In the case of division of two images, the image dimensions must be the same for both images.

Arguments

imgDiv with no arguments divides image pairs from *stdin*. If a numeric is specified on the command line, this is interpreted as a constant to divide by. Otherwise, this argument is interpreted as the filename of an image to load.

Treatment of Valid Region

The valid region is not changed by this operator, but is ignored for the purposes of processing. The process is applied to the whole image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are obtained during processing and are propagated with the output image.

See Also

imgAbs, imgAdd, imgMul, imgSub, imgNeg.

imgEdgeSynth/imgEdgeSynth2

Name

imgEdgeSynth—Canny two-level edge synthesis operation.

Syntax

imgEdgeSynth [low_thresh high_thresh]
imgEdgeSynth2 [low_thresh high_thresh [filename]]

Image Types

imgEdgeSynth

stdin	Float format
stdout	Bit format (pbm)

imgEdgeSynth2

stdin	Bit format (pbm), Float format
stdout	Bit format (pbm)
file	Float format

Description

This command carries out J. Canny's two level edge thresholding and synthesis operation. Two forms of the tool are offered. For imgEdgeSynth2, the input to this process is a complete edge-map image and a value image. The edge-map is a PBM image containing black pixels (pixels set to 1) for all conceivable edge points i.e. prior to validation. The value image is used to validate the edge-map according to the thresholding mechanism described below. This input image is floating point and should contain the magnitude of the gradient of the smoothed original image to implement Canny's scheme. Two thresholds are applied to the value image: an upper and a lower one. All edge-map information that overlaps the parts of the value image above the upper threshold is copied into the output image. Further, these line segments are then extended according to the edge-map while they overlap part of the value image that is above the lower threshold. This produces continuous lines that are not easily broken by image noise. If the threshold values are equal, straightforward threshold validation results. The edge-map and value images must be the same size.

For imgEdgeSynth, the process is the same, but the input is different. Instead of having a bit image showing edges and a float image to validate it, the two pieces of information are combined and only one float image is needed. This has the same form as that output by imgCanny and consists of a floating point edge map in which non-zero pixels indicate edge points and these have a value which indicates the edge strength.

Arguments

The low and high thresholds are given on the command line, followed by the filename of the floating point image to load and apply thresholding to. If the filename is not specified, then the function reads the floating point image from *stdin* after it has read the PBM edge-map image.

Treatment of Valid Region

Valid regions are not supported with this image type.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are not supported with this image type.

See Also

imgMarrHil, imgCanny, imgZeros.

imgFFTBandPass

Name

imgFFTBandPass—Fourier Gaussian bandpass filter.

Syntax

imgFFTBandPass [freq [bandwidth]]

Image Types

stdin	Complex format
stdout	Complex format

Description

This operation carries out bandpass filtering on an image by multiplication in the frequency domain. The bandpass function is implemented using a frequency domain Gaussian function, shifted so that the peak is at the given frequency. The function is symmetrical about this frequency and has unity gain at the peak. The half-height bandwidth of the function can be specified in octaves. Note that the DC gain is non-zero and is significant for bandwidths greater than one octave.

Arguments

The centre frequency of the bandpass function is specified in radians per pixel. The default value is $\pi/2$. The bandwidth is specified in octaves and the default value is one octave at half height.

Treatment of Valid Region

The valid region is ignored, but propagated to the output image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values in the input image are not used and are not propagated to the output image.

See Also

imgFFTDOG, imgFFTGabor, imgFFTHigh, imgFFTLow, imgFFTSmooth.

imgFFTDOG

Name

imgFFTDOG—Fourier difference of Gaussian filter.

Syntax

imgFFTDOG [rc rs [kc ks]]

Image Types

stdin	Complex format
stdout	Complex format

Description

This operation carries out a bandpass filtering operation by multiplication in the frequency domain. The bandpass function is a difference of Gaussian function. In the spatial domain, this corresponds to a centre-surround kernel comprising a large radius negative Gaussian added to a small radius positive Gaussian, centred at the same position. The half-height radii of the each Gaussian can be specified separately, as can the peak value (gain constant).

Arguments

The first two arguments specify the $e^{-0.5}$ radii for centre and surround parts. These default to 1.0 and 3.0 pixels respectively. The centre and surround gain constants can also be specified. If these constants are missing, the default action is to calculate them from the radii so that the filtering operation has zero gain at DC (the positive and negative portions cancel) and the gain at the frequency domain maxima is unity.

Treatment of Valid Region

The valid region is ignored, but propagated to the output image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values in the input image are not used and are not propagated to the output image.

See Also

imgFFTBandPass, imgFFTGabor, imgFFTHigh, imgFFTLow, imgFFTSmooth.

imgFFTGabor

Name

imgFFTGabor—Fourier directional Gabor filter.

Syntax

imgFFTGabor [orient [freq [incr [bandwidth]]]]

Image Types

stdin	Complex format
stdout	Complex format

Description

This processing tool filters the image with a Gabor filter. In the spatial domain this filter has a Gaussian profile along one axis and a Gabor (Gaussian enveloped sine wave) profile along the other axis. The orientation of the filter can be specified on the command line as can its centre frequency and the bandwidth of the Gaussian profile. The command actually produces two complex images for every input image. These two images are the results of applying filters with first cosine (even) and then sine (odd) phase. Note that with bandwidths greater than about 1.0, the DC response of the even Gabor filter becomes significant. The orientation of the filter can be made to auto-increment with each new input image.

Arguments

The orientation parameter specifies the spatial orientation of the filter in degrees. Zero degrees extracts horizontal spatial frequencies around the filter centre frequency. This is the default. The frequency parameter specifies the centre frequency of the filter in radians per pixel and defaults to $\pi/2$. The increment parameter, if specified, causes the orientation to be auto-incremented by this angle (in degrees) for each new input image. The default is not to auto-increment. The bandwidth is the Gabor Gaussian half-height bandwidth, affecting both dimensions of the filter. This parameter is expressed in octaves and defaults to 1.0.

Treatment of Valid Region

The valid region is ignored, but propagated to the output image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values in the input image are not used and are not propagated to the output image.

See Also

imgFFTBandPass, imgFFTDOG, imgFFTHigh, imgFFTLow,
imgFFTSmooth.

imgFFTHigh

Name

imgFFTHigh—Fourier low cut, high-pass filter.

Syntax

imgFFTHigh [cutoff]

Image Types

stdin	Complex format
stdout	Complex format

Description

This command implements a high-pass filter by operations in the frequency domain. The filter is a non-directional box profile filter that passes all spatial frequencies above the cutoff threshold and blocks all frequencies below this. It has unity gain above the threshold frequency and zero gain below it.

Arguments

The argument selects the frequency at which the cutoff will take place in radians per pixel. If this argument is missing, the cutoff is set to the frequency that imgFFTSmooth gives a half height response with a smoothing constant of 1.0 (1.17741 rad/pix).

Treatment of Valid Region

The valid region is ignored, but propagated to the output image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values in the input image are not used and are not propagated to the output image.

See Also

imgFFTBandPass, imgFFTDOG, imgFFTGabor, imgFFTLow, imgFFTSmooth.

imgFFTLow

Name

imgFFTLow—Fourier high cut, low-pass filter.

Syntax

imgFFTLow [cutoff]

Image Types

stdin	Complex format
stdout	Complex format

Description

This command implements a low-pass filter by operations in the frequency domain. The filter is a non-directional box profile filter that passes all spatial frequencies below the cutoff threshold and blocks all frequencies above this. It has unity gain below the threshold frequency and zero gain above it.

Arguments

The argument selects the frequency at which the cutoff will take place in radians per pixel. If this argument is missing, the cutoff is set to the frequency that imgFFTSmooth gives a half height response with a smoothing constant of 1.0 (1.17741 rad/pix).

Treatment of Valid Region

The valid region is ignored, but propagated to the output image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values in the input image are not used and are not propagated to the output image.

See Also

imgFFTBandPass, imgFFTDOG, imgFFTGabor, imgFFTHigh, imgFFTSmooth.

imgFFTSmooth

Name

imgFFTSmooth—Fourier Gaussian smoothing filter.

Syntax

imgFFTSmooth [sigma]

Image Types

stdin	Complex format
stdout	Complex format

Description

This function carries out Gaussian smoothing in the frequency domain. The value of *sigma* is the Gaussian width parameter. Larger values will lead to more smoothing and a lower frequency domain roll-off.

Arguments

When no parameter is specified, *sigma* defaults to 1.0 corresponding to a frequency at half height of 1.17741 rad/pix.

Treatment of Valid Region

The valid region is ignored, but propagated to the output image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values in the input image are not used and are not propagated to the output image.

See Also

imgFFTBandPass, imgFFTDOG, imgFFTGabor, imgFFTHigh, imgFFTLow.

imgFlip

Name

imgFlip—Reflect image horizontally or vertically.

Syntax

imgFlip [direction]

Image Types

stdin	Byte format (pgm)
stdout	Byte format (pgm)

Description

This command flips an image by reflecting it around the centre vertical or horizontal axis.

Arguments

The direction argument can be ‘x’, ‘y’ or ‘xy’, for flipping left-right, top-bottom or both.

Arguments

See Also

imgReorder.

imgFlt2ToCpx

Name

imgFlt2ToCpx—Convert two floating point images into complex format.

Syntax

imgFlt2ToCpx

Image Types

stdin	Float format
stdout	Complex format

Description

imgFlt2ToCpx converts two successive floating point images into one complex format image. The first input image is the real part and the second is the imaginary part. The input images must have the same dimensions.

Treatment of Valid Region

The valid region from the floating point image representing the real part is propagated to the output complex image. The second valid region is thrown away. The valid region information is ignored for the purposes of copying the data.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values in the input image are not used and are not propagated to the output image.

See Also

imgPnmToFlt, imgFltToPgm, imgPpmToColour, imgPpmToFlt3, imgFlt3ToPpm, imgCpxToFlt2.

imgFlt3ToPpm

Name

imgFlt3ToPpm—Convert three floating point images into PbmPlus type PPM images.

Syntax

imgFlt3ToPgm [min max]

Image Types

stdin	Float format
stdout	Rgb format (ppm)

Description

imgFlt3ToPpm converts floating point images into PPM colour rgb formatted images. The values are scaled into the range 0 to 255 with the use of minimum and maximum values specified in the input image header or from the command line. The floating point pixel values are scaled and then offset to convert them into the byte range using the same method as that used by imgFltToPgm. Three floating point images representing the red, green and blue components are read from *stdin* for each PPM image output by the program.

Arguments

The optional *min* and *max* arguments override the minimum and maximum values specified in the image header.

Treatment of Valid Region

The input valid region is ignored and there is no output valid region since this is not defined for a PPM image file.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are used to set the scaling and offset values used in the mapping of floating point values into byte values as defined above. The range values used in the mapping are those of the widest composite range derived from the separate float images. There is no minimum and maximum value data accompanying the output image.

See Also

imgPnmToFlt, imgFltToPgm, imgPpmToColour, imgPpmToFlt3, imgFlt2ToCpx, imgCpxToFlt2.

imgFltToPgm

Name

imgFltToPgm—Convert floating point images into PbmPlus type PGM images.

Syntax

imgFltToPgm [min max]

Image Types

stdin	Float format
stdout	Byte format (pgm)

Description

imgFltToPgm converts floating point images into PGM monochrome byte formatted images. The values are scaled into the range 0 to 255 with the use of minimum and maximum values calculated by the program, specified in the input image header or specified on the command line. The floating point pixel values are scaled and then offset to convert them into the byte range. If the *min* and *max* values are both positive, the input range [0.0,*max*] maps to the output range [0,255] and *min* is ignored. If the *min* value is negative and the *max* value is positive, the input range [-*val*,*val*] is mapped into the range [0,255] such that a floating point 0.0 generates a byte value of 128, where *val* is the "abs largest" value of *min* and *max*. If the *min* and *max* values are both negative, the input range [*min*,0.0] maps to the output range [0,255] and *max* is ignored. In this way, wholly positive or wholly negative images are mapped to the full byte range, but those that vary around zero are mapped into a representation varying around 128. This tool prints out the values of *min* and *max* that it uses.

Arguments

The optional *min* and *max* arguments override the minimum and maximum values specified in the image header. If these values are both zero in the header then imgFltToPgm calculates them itself.

Treatment of Valid Region

The input valid region is ignored and there is no output valid region since this is not defined for a PGM image file.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are used to set the scaling and offset values used in the mapping of floating point values into byte values as defined above. There is no minimum and maximum value data accompanying the output image.

See Also

imgPnmToFlt, imgPpmToColour, imgPpmToFlt3, imgFlt3ToPpm, imgFlt2ToCpx, imgCpxToFlt2.

imgForwardFFT

Name

imgForwardFFT—Calculate the forward fast Fourier transform of an image.

Syntax

imgForwardFFT

Image Types

stdin	Float format
stdout	Complex format

Description

imgForwardFFT calculates the fast Fourier transform of an image using the method described in "Numerical Recipes in C". The input image must be square and have a dimension that is a power of two. This command is often used for convolution by frequency domain multiplication.

Treatment of Valid Region

The valid region information is propagated unchanged by this function even though the output image is a frequency domain image. This is so that valid region information will remain across Fourier convolutions.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are not set by this function.

See Also

imgCpxMul, imgInverseFFT.

imgFourierPad

Name

imgFourierPad—Convert a convolution kernel into a form useful for Fourier processing.

Syntax

imgFouierPad size

Image Types

stdin	Float format, Double format
stdout	Float format

Description

This tool takes an arbitary-sized image in either float or double format and pastes it onto a blank (=0.0) image of the size specified on the command line so that the centre pixel is at location (0,0). This is used to produce Fourier kernels for use with imgCpxMul from space-domain kernels. Pass the (small) space domain kernel through this tool to make it the correct size (the size of the image to be processed), then Fourier transform it using imgForwardFFT. Save the resulting Fourier kernel to a file. Next process the image by using imgForwardFFT, imgCpxMul and imgInverseFFT in succession, with the Fourier kernel file as an argument to imgCpxMul. See imgConvolve to find out how to make a space-domain kernel.

Arguments

The size argument specifies a square image and must be a power of two.

Treatment of Valid Region

The valid region information is propagated unchanged by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are re-calculated by this function.

See Also

imgForwardFFT, imgCpxMul, imgInverseFFT.

imgFoveate

Name

imgFoveate—Foveally smooth image with a Gaussian kernel.

Syntax

imgFoveate [incr [sigma [x y]]]

Image Types

stdin	Float format
stdout	Float format

Description

This command smooths an image by different amounts in different places so that one region of the image (the foveal centre) is smoothed very little, if at all, and the amount of smoothing increases with radial distance from this point. The process uses Gaussian smoothing with variable width factor. This process runs incredibly slowly because the kernel must be recalculated for each image point.

Arguments

The parameter *sigma* sets the Gaussian width at the foveal centre and defaults to 0.0 (no smoothing). The parameter *incr* sets the smoothing factor increment with distance from the foveal centre. This factor is multiplied by the distance in pixel units and added to the centre *sigma* value to get the smoothing factor at any point. The parameters x and y optionally set the foveal centre to be other than at the image centre.

Treatment of Valid Region

The valid region is not altered by this command. The smoothing kernel is progressively truncated as the image boundaries are reached.

Treatment of Minimum and Maximum Values

Minimum and maximum values are recalculated during this process.

See Also

imgSmooth, imgFFTSmooth.

imgGrad

Name

imgGrad—Calculate image gradient vector at each pixel.

Syntax

imgGrad

Image Types

stdin	Float format
stdout	Complex format

Description

This command calculates the gradient of an image, yielding a two dimensional vector, output as a complex image. The real part corresponds to the derivative in the x direction. The derivative in the y direction is defined as positive up the image. A 2×2 kernel is used with all four values used to compute the partial derivatives in the two orthogonal directions. The result pixel for each point is positioned in registration with the top left kernel point.

Treatment of Valid Region

Processing only takes place within the input valid region. The Output valid region is one pixel smaller in width and height than the input region.

Treatment of Minimum and Maximum Values

Minimum and maximum values are not set by this command.

See Also

imgAngle, imgMagnitude, imgDeriv, imgLaplacian.

imgGrating

Name

imgGrating—Generate a sine-wave grating for test purposes.

Syntax

imgGrating [frequency [orientation [width height [contrast [mean [phase]]]]]]

Image Types

stdin	None
stdout	Float format

Description

This command produces a sine wave grating of a single frequency with parameters as specified.

Arguments

Grating frequency is specified in radians per pixel, orientation in degrees from the horizontal and phase in degrees from zero sine-wave phase, with increasing phase causing a vertical grating to move left. The mean and contrast arguments set the additive zero offset and sine multiplier constants respectively. The default action is to generate a 256×256 image with a vertically oriented grating of frequency $\pi/2$ radians per pixel (4 pixels per cycle) of phase 0.0, mean value 0.0 and contrast 1.0.

Treatment of Valid Region

The valid region is set to the whole of the image.

Treatment of Minimum and Maximum Values

Minimum and maximum values are set to $mean \pm contrast$.

See Also

imgImpulse.

imgImage

Name

imgImage—Generate a blank Image.

Syntax

imgImage [width height] [value]

Image Types

stdin	None
stdout	Float format

Description

This command produces a blank image of the specified size and value.

Arguments

With no arguments, the default operation is to generate a 256×256 image with all pixels set to 0.0. If one argument is given then pixel values are set to this value. If two or more arguments are given then the first two are taken to be the size of the image and the third (if present) is the pixel value.

Treatment of Valid Region

The valid region is set to the whole of the image.

Treatment of Minimum and Maximum Values

Minimum and maximum values are both set to the pixel value provided.

See Also

imgGrating, imgImpulse.

imgImpulse

Name

imgImpulse—Generate a single impulse for test purposes.

Syntax

imgImpulse [width height [x y [value]]]

Image Types

stdin	None
stdout	Float format

Description

This command produces an image containing a single non-zero value for test purposes: for example, producing the impulse response of a linear operator. The size of the image to produce, position of the impulse and value of the impulse can be set from the command line.

Arguments

With no arguments, the default operation is to generate a 256×256 image with the centre pixel set to 1.0.

Treatment of Valid Region

The valid region is set to the whole of the image.

Treatment of Minimum and Maximum Values

Minimum and maximum values are set to 0.0 and *value* as appropriate.

See Also

imgGrating.

imgInfo

Name

imgInfo—Show information about the current image.

Syntax

imgInfo

Image Types

stdin	Any format
stdout	None

Description

imgInfo prints information about the input image. The information given is the image type, size, valid area and minimum and maximum values. For the case of Bit, Byte and Rgb images, the valid region is given as the whole image area and the minimum and maximum values are given as [0.0,1.0], [0.0,255.0] and [0.0,255.0] respectively.

See Also

imgAscii, imgPrint, imgStats.

imgInsert

Name

imgInsert—Insert images from sequenced files into the image stream.

Syntax

imgInsert basename [start [incr [count [insert_start [insert_incr]]]]]

Image Types

stdin	Any format
stdout	Any format
file	Any format

Description

imgInsert loads the image files specified by the command line parameters and inserts them in the stream from *stdin* to *stdout*. The image files form a sequence with three digits in the filename indicating the sequence number. Files are loaded in ascending order starting from the file with sequence number *start* and incrementing by *incr* each time. At most *count* files are loaded. The insert parameters indicate where in the stream the files are to be inserted. Images loaded from files are interspersed with input images as specified until either image source is exhausted. Following this, images are output consecutively from the non-exhausted source.

Arguments

The filename argument specifies the basename of the file sequence. If a dot extension is specified, then the three digit sequence number will be inserted after the basename and before this extension. With no further command options, the default action is to load the file exactly as given, without a sequence number. If the *start* parameter is given, this sets the sequence number of the first file to be loaded. With no further options, imgInsert loads files consecutively from this number until no further file is found. If an increment is specified then this is used to generate further sequence numbers. The parameter *incr* can be negative, but note that a file sequence number cannot be negative. The *count* parameter specifies the number of files to load.

The output image stream slots are numbered from zero onwards. If *insert_start* is set to zero then the first output image will come from a file. If the parameter is one, then the first image will come from *stdin* and the second from a file. The increment option indicates how often the images from the file source are to be inserted in the output stream. The default values of *insert_start* and em insert_incr are 1 and 2 respectively. This causes the first image to be taken from *stdin* and the next from the first sequence file and subsequent images to be drawn from the two sources in alternation.

Treatment of Valid Region

The valid region information is propagated unaltered by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCombine, imgCombineSeq, imgRepeat, imgSelect, imgSplit, imgSplitSeq, imgTee.

imgInverseFFT

Name

imgInverseFFT—Calculate the inverse fast Fourier transform of an image.

Syntax

imgInverseFFT

Image Types

stdin	Complex format
stdout	Float format

Description

imgInverseFFT calculates the inverse fast Fourier transform of an image using the method described in "Numerical Recipes in C". The input image must be square and have a dimension that is a power of two. This command is often used for convolution by frequency domain multiplication.

Treatment of Valid Region

The valid region information is propagated unchanged by this function even though the input image is a frequency domain image. This is so that valid region information passed through imgForwardFFT will remain across Fourier convolutions.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are not set by this function.

See Also

imgCpxMul, imgForwardFFT.

imgLaplacian

Name

imgLaplacian—Calculate image Laplacian at each pixel.

Syntax

imgLaplacian

Image Types

stdin	Float format
stdout	Float format

Description

This command calculates the Laplacian of an image, yielding a floating point result. A 3×3 kernel is used and the result pixel for each point is positioned in registration with the centre kernel point.

Treatment of Valid Region

Processing only takes place within the input valid region. The Output valid region is two pixels smaller in width and height than the input region.

Treatment of Minimum and Maximum Values

Minimum and maximum values are calculated from the results of this process applied within the valid input region.

See Also

imgAngle, imgMagnitude, imgDeriv, imgGrad.

imgMagnify

Name

imgMagnify—Increase pixel size of input images.

Syntax

imgMagnify [scale]
imgMagnify [x_scale y_scale]

Image Types

stdin	Any format
stdout	Any format

Description

imgMagnify increases the size of an image by replicating each pixel an integer number of times horizontally and vertically. The effect can be reversed by using the command imgSubSample.

Arguments

imgMagnify with no arguments scales the picture by a factor of two. When one argument is given, both x and y-axes are scaled by the same amount. Two arguments specify the scalings separately.

Treatment of Valid Region

The valid region information from the input image is scaled up according to the arguments given to give the output valid region.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCrop, imgReorder, imgSubSample.

imgMagnitude

Name

imgMagnitude—Calculate the magnitude of a pixel vector.

Syntax

imgMagnitude [number]

Image Types

stdin	Float format, Complex format, Polar format
stdout	Float format

Description

This command calculates the magnitude of an n-dimensional vector defined at each pixel. If the input image is of complex form, then the magnitude of the complex quantity is calculated by geometric combination. If the input image is of polar form, the magnitude information is extracted directly. In general, this command works by squaring the components of the input, adding them together across dimensions and square rooting the result. The command can carry out this operation across a number of input images to handle large numbers of dimensions. Thus a 2D vector can be represented by one complex image or two floating point images.

Arguments

The command-line argument specifies the number of input images to combine in forming the geometric sum. The default is one. The images may be of mixed type.

Treatment of Valid Region

The output valid region is set to the valid region defined for the first image in each group.

Treatment of Minimum and Maximum Values

The minimum output value is set to zero as this is generally more useful. The maximum output value is calculated during the magnitude process.

See Also

imgAngle, imgDeriv, imgGrad, imgLaplacian.

imgMarrHil

Name

imgMarrHil—Marr-Hildreth type edge operator.

Syntax

imgMarrHil

Image Types

stdin	Float format
stdout	Float format

Description

This command essentially takes the Laplacian of the image values. It forms part of the edge detection process proposed by D. Marr and E. Hildreth. The process needs to be preceeded by Gaussian smoothing using imgSmooth. Zero-crossing detection is included after the Laplacian stage to locate edge points. A 3×3 kernel is used and the output points are in registration with the kernel centre for any kernel position. The output is an image in which all non-edge pixels have value 0.0 and edge pixels have a positive value which is equal to the magnitude of the image gradient at that point. This output is suitable for input into imgEdgeSynth.

Treatment of Valid Region

The output valid region is set to the valid region defined for the input image with an adjustment for the kernel size.

Treatment of Minimum and Maximum Values

The minimum and maximum values are calculated during processing.

See Also

imgCanny, imgZeros, imgEdgeSynth, imgLaplacian.

imgMax

Name

imgMax—Find maximum pixel values over N images.

Syntax

imgMax [N]

Image Types

stdin	Float format
stdout	Float format

Description

imgMax reads in floating poit images in batches of N. It then computes the maximum pixel values at each point accross this set of images and places the result in the output image.

Arguments

The optiional argument specifies the number of images to be max'ed together. The default is 2.

Treatment of Valid Region

The output valid region is the same as valid region for the first image of each set.

Treatment of Minimum and Maximum Values

The minimum and maximum values are calculated during processing.

See Also

imgAdd, imgMul, imgSub, imgNeg, imgAbs, imgSub.

imgMul

Name

imgMul—Multiply images or multiply image by a constant.

Syntax

imgMul [constant]
imgMul [filename]

Image Types

stdin	Float format
stdout	Float format
file	Float format

Description

This process performs a pixel-wise multiplication of images. Each input image is processed in turn and multiplication is performed in one of three ways: Either all pixel values are multiplied by a constant provided on the command line, or the image file named on the command line is multiplied in a pixel-wise manner, or else the next input image is multiplied in a pixel-wise manner. In the case of multiplication of two images, the image dimensions must be the same for both images.

Arguments

imgMul with no arguments divides image pairs from *stdin*. If a numeric is specified on the command line, this is interpreted as a constant to multiply by. Otherwise, this argument is interpreted as the filename of an image to load.

Treatment of Valid Region

The valid region is not changed by this operator, but is ignored for the purposes of processing. The process is applied to the whole image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are obtained during processing and are propagated with the output image.

See Also

imgAbs, imgAdd, imgDiv, imgSub, imgNeg.

imgNeg

Name

imgNeg—Negate image values.

Syntax

imgNeg

Image Types

stdin	Float format
stdout	Float format

Description

For each image in the input sequence, imgNeg switches the sign of the floating point value on a per-pixel basis.

Treatment of Valid Region

The valid region is not changed by this operator, but is ignored for the purposes of processing. The process is applied to the whole image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are obtained during processing and are propagated with the output image.

See Also

imgAbs, imgAdd, imgMul, imgDiv, imgSub.

imgNoise

Name

imgNoise—Add Gaussian random noise to a floating point image.

Syntax

imgNoise [std-deviation]

Image Types

stdin	Float format
stdout	Float format

Description

imgNoise adds random noise with a Gaussian distribution to the floating point images that pass through this operator.

Arguments

The optional argument specifies the noise standard deviation. This value defaults to 1.0.

Treatment of Valid Region

The valid region is passed on without change.

Treatment of Minimum and Maximum Values

New minimum and maximum values are calculated over the whole image.

See Also

imgAdd, imgMul, imgDiv, imgSub.

imgNot

Name

imgNot—Perform logical NOT of a bit image.

Syntax

imgNot

Image Types

stdin	Bit format (pbm)
stdout	Bit format (pbm)

Description

This process performs a pixel-wise logical NOT. Each input image is processed in turn and the logical inversion operation is performed.

Treatment of Valid Region

Valid regions are not supported with this image type.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are not supported with this image type.

See Also

imgAnd, imgOr, imgXor.

imgOr

Name

imgOr—Perform logical OR between bit images.

Syntax

imgOr [filename]

Image Types

stdin	Bit format (pbm)
stdout	Bit format (pbm)
file	Bit format (pbm)

Description

This process performs a pixel-wise logical OR of images. Each input image is processed in turn and the operation is performed in one of two ways: Either the image file named on the command line is ORed with the input image in a pixel-wise manner, or else the next input image is ORed in a pixel-wise manner. The image dimensions must be the same for both images.

Arguments

imgOr with no arguments operates on image pairs from *stdin*. Otherwise, the argument is interpreted as the filename of an image to load.

Treatment of Valid Region

Valid regions are not supported with this image type.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are not supported with this image type.

See Also

imgAnd, imgXor, imgNot.

imgPnmToFlt

Name

imgPnmToFlt—Convert PbmPlus type images into floating point format.

Syntax

imgPnmToFlt

Image Types

stdin	Bit format (pbm), Byte format (pgm) or Rgb format (ppm)
stdout	Float format

Description

imgPnmToFlt converts PbmPlus format images into floating point images for use with other processing tools. The PBM, PGM and PPM formats are supported. PBM bit image pixel values are converted into the values 0.0 and 1.0, where 1.0 represents *white*. PGM images are converted into the range 0.0 to 255.0 and PPM colour pixels are converted into a monochrome representation in the range 0.0 to 255.0. The colour conversion formula is: $y = (0.299r + 0.587g + 0.114b)$.

Treatment of Valid Region

The output valid region is set equal to the whole image area.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are set to 0.0 and 255.0 respectively for the PBM and PGM input types and to 0.0 and 1.0 for the PBM image type. Fixed ranges are specified so that imgPnmToFlt followed by imgFltToPgm is an identity transformation.

See Also

imgFltToPgm, imgPpmToColour, imgPpmToFlt3, imgFlt3ToPpm, imgFlt2ToCpx, imgCpxToFlt2.

imgPower

Name

imgPower—Raise image values to a constant power.

Syntax

imgPower [power]

Image Types

stdin	Float format
stdout	Float format

Description

imgPower raises all image values to the constant power provided on the command line. This can be used to implement gamma conditioning. If the input image contains any negative pixel values then raising to a non-integer power will result in a flood of domain errors. Note that this may sometimes happen if the input image has been filtered using Fourier transform processing, even when the space-domain convolution would produce wholly positive results. This is due to arithmetic accuracy during FFTs when the pixel values are near zero.

Arguments

imgPower with no arguments is raises to the power 1.0. The power value is a real number.

Treatment of Valid Region

The output valid region is set equal to the input valid region.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are calculated across the whole image.

See Also

imgAdd, imgSub, imgMul, imgDiv, imgNeg.

imgPpmToColour

Name

imgPpmToColour—Convert PPM image into two orthogonal colour components.

Syntax

imgPpmToColour

Image Types

stdin	Rgb format (ppm)
stdout	Float format

Description

This command calculates two orthogonal colour components for each pixel and produces two floating point images for every PPM input image. The standard conversion from colour to monochrome carried out by imgPnmToFlt is achieved by dot producting each image RGB value by the vector [0.299 0.587 0.114]. imgPpmToColour dot products each RGB value by [0.488 -0.456 -0.032] to give a colour component in the red-green direction and by [-0.181 -0.355 0.536] to give a colour component in the blue-yellow direction. These two components have the same scale as the monochrome component and are orthogonal to it, and to each other.

Treatment of Valid Region

The valid region is not altered by this command.

Treatment of Minimum and Maximum Values

Minimum and maximum values are recalculated during this process.

See Also

imgFltToPgm, imgPnmToFlt, imgPpmToFlt3, imgFlt3ToPpm, imgFlt2ToCpx, imgCpxToFlt2.

imgPpmToFlt3

Name

imgPpmToFlt3—Convert PPM colour images into floating point format.

Syntax

imgPpmToFlt3

Image Types

stdin	Rgb format (ppm)
stdout	Float format

Description

imgPpmToFlt3 converts PbmPlus format PPM colour images into floating point format. Three floating point images are generated for each input image giving the red, green and blue components of each pixel.

Treatment of Valid Region

The output valid region is set equal to the whole image area.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are set to 0.0 and 255.0 respectively.

See Also

imgFltToPgm, imgPnmToFlt, imgPpmToColour, imgFlt3ToPpm, imgFlt2ToCpx, imgCpxToFlt2.

imgPrint

Name

imgPrint—Print contents of an image sub-region.

Syntax

imgPrint [x y [width height]]

Image Types

stdin	Any format
stdout	None

Description

imgPrint displays the pixel values of a subregion of each input image in a convenient human-readable format. All image types are supported. The printing format varies with the image type. Information about the coordinates of the pixel rows and at least the first column is also provided. If no command arguments are given, the (7×7) pixels starting from the coordinates of the top left corner of the valid image region are printed.

Arguments

The optional x and y arguments specify the top left corner of the region to be printed and the *width* and *height* arguments can be used to change the size of the region. Information will only be printed for pixels within the image boundaries.

See Also

imgAscii, imgInfo, imgStats.

imgRankFilter

Name

imgRankFilter—Perform rank-value filtering on a floating point image.

Syntax

imgRankFilter [rank]

Image Types

stdin	Float format
stdout	Float format

Description

imgRankFilter performs rank-value filtering by first sorting the nine pixel values in each 3×3 neighbourhood into ascending order in an array. From this array, the value with index equal to *rank* is picked. This is then put back into the image as the centre pixel. The array slots are numbered from 1–9 and by choosing appropriate values of *rank*, such an operation can be used to perform dilation (*rank*=9), erosion (*rank*=1) or median filtering (*rank*=5). Median filtering is useful in removing spurious isolated pixel noise.

Arguments

The optional argument indicates the array slot and this defaults to 5 (median filtering).

Treatment of Valid Region

The valid region is reduced by two in both directions.

Treatment of Minimum and Maximum Values

Minimum and maximum values are propagated unaltered.

See Also

imgSmooth, imgConvolve.

imgRectify

Name

imgRectify—Half-wave rectify all image values.

Syntax

imgRectify [compare]

Image Types

stdin	Float format
stdout	Float format

Description

This command sets to zero all negative image values. The effect is to take only positive values and therefore carry out half-wave rectification.

Arguments

The optional argument sets the commands concept of negative image values. Essentially, pixel values less than *compare* are set to zero. This argument defaults to zero to give the conventional rectifying action.

Treatment of Valid Region

The valid region is preserved by this function. The process is applied to the whole image.

Treatment of Minimum and Maximum Values

Minimum and maximum values are recalculated during this process.

See Also

imgAbs, imgThreshZero.

imgReorder

Name

imgReorder—Re-arrange floating point image for Fourier transform viewing.

Syntax

imgReorder

Image Types

stdin	Float format
stdout	Float format

Description

The image data produced by the imgForwardFFT command has the DC term at the image origin and has positive frequency points in the top left quadrant of the image. This command reorders floating point data from imgMagnitude for instance, such that the origin is in the centre of the image and the frequency v axis increases with decreasing y. This is the form that the transform commonly used for display purposes.

Treatment of Valid Region

The valid region information is propagated unaltered by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCrop, imgMagnify, imgSubSample.

imgRepeat

Name

imgRepeat—Produces a repeated sequence of each input image.

Syntax

imgRepeat [times]

Image Types

stdin	Any format
stdout	Any format

Description

imgRepeat loads an image from *stdin*, outputs it a number of times to *stdout* and then moves on to the next input image. This is used when the same image is to be processed a number of different ways by one image processing tool. The process to be applied is altered between each input image.

Arguments

The optional parameter specifies the number of times to repeat each image. The default is twice.

Treatment of Valid Region

The valid region information is propagated unaltered by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCombine, imgCombineSeq, imgInsert, imgSelect, imgSplit,
imgSplitSeq, imgTee.

imgSelect

Name

imgSelect—Pick out a set of images from the stream.

Syntax

imgSelect [start [incr [count]]]

Image Types

stdin	Any format
stdout	Any format

Description

imgSelect passes a subset of the images presented to it from *stdin*. This operation allows unwanted images to be removed from the image stream. An example might be in conjunction with the command imgCpxToFlt. This command generates two images from every one received. The output image pairs are alternately real and imaginary data. If real data only is required for subsequent processing, the imgSelect command can extract these alternate images, yielding a consecutive stream of real data.

Arguments

The *start, incr* and *count* parameters specify the position and number of images to pass to *stdout*. The default operation is to pick out alternate images starting from the first one. This corresponds to parameters of 0 and 2 respectively. If the last parameter is missing, the operation continues as long as the input stream lasts.

Treatment of Valid Region

The valid region information is propagated unaltered by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCombine, imgCombineSeq, imgInsert, imgRepeat, imgSplit, imgSplitSeq, imgTee.

imgSetMinMax

Name

imgSetMinMax—Set minimum and maximum value fields of a floating point image.

Syntax

imgSetMinMax [min max]

Image Types

stdin	Float format
stdout	Float format

Description

This tool sets the minimum and maximum value fields that are propogated within the header of a floating point image. Most tools automatically calculate these values. This tool allows one to set them manually or else have them automatically calculated from all the pixel values present within the valid region.

Arguments

Minimum and maximum values are optionally specified.

Treatment of Valid Region

The output valid region is set equal to the input valid region.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are either set to the values supplied or else calculated over the input valid region.

See Also

imgFltToPgm.

imgSmooth

Name

imgSmooth—Smooth image with a Gaussian kernel.

Syntax

imgSmooth [sigma]

Image Types

stdin	Float format
stdout	Float format

Description

This function carries out Gaussian smoothing. The value of *sigma* is the Gaussian width parameter. The kernel size depends on the this parameter, and since this tool uses a two part spatial convolution, the run time will increase linearly with an increase in this parameter. The kernel size is always odd.

Arguments

When no parameter is specified, *sigma* defaults to 1.0. Otherwise, the value can range from 0.1 to 30.0.

Treatment of Valid Region

The valid region is reduced in size by this command. The amount of reduction depends on the kernel size.

Treatment of Minimum and Maximum Values

Minimum and maximum values are recalculated during this process.

See Also

imgFoveate, imgMarrHil, imgCanny, imgZeros, imgFFTSmooth.

imgSplit

Name

imgSplit—Write images from the input stream to files.

Syntax

imgSplit file1 [file2 [...]]

Image Types

stdin	Any format
stdout	None
file	Any format

Description

imgSplit writes the files presented to *stdin* to the list of filenames specified on the command line. If all the filenames are used up, the command throws away further input.

Arguments

The arguments consist of a list of filenames to write consecutive images to. At least one must be present.

Treatment of Valid Region

The valid region information is propagated unaltered by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCombine, imgCombineSeq, imgInsert, imgRepeat, imgSelect, imgSplitSeq, imgTee.

imgSplitSeq

Name

imgSplitSeq—Write images from the input stream to sequenced filenames.

Syntax

imgSplitSeq basename [start [incr [count]]]

Image Types

stdin	Any format
stdout	None
file	Any format

Description

imgSplitSeq writes the files presented to *stdin* to the sequence of filenames specified by the command line arguments. The image files form a sequence with three digits in the filename indicating the sequence number. Files are written in ascending order starting from the file with sequence number *start* and incrementing by *incr* each time. At most *count* files are written. If a smaller number of images are written than are presented to the input, further input is thrown away.

Arguments

The filename argument specifies the basename of the file sequence. If a dot extension is specified, then the three digit sequence number will be inserted after the basename and before this extension. With no further command options, the default action is to write the file with the name exactly as given, without a sequence number. If the *start* parameter is given, this sets the sequence number of the first file. With no further options, imgSplitSeq writes files consecutively from this number until no further input is present. If an increment is specified then this is used to generate further sequence numbers. The parameter *incr* can be negative, but note that a file sequence number cannot be negative. The final parameter specifies the maximum number of files to write.

Treatment of Valid Region

The valid region information is propagated unaltered by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCombine, imgCombineSeq, imgInsert, imgRepeat, imgSelect, imgSplit, imgTee.

imgStats

Name

imgStats—Calculate statistical information.

Syntax

imgStats

Image Types

stdin	Float format
stdout	None

Description

imgStats prints statistical information about input images. The mean, root-mean-square and standard deviation of all the pixel values are computed and printed.

See Also

imgAscii, imgInfo, imgPrint.

imgSub

Name

imgSub—Subtract images or subtract constant from image.

Syntax

imgSub [constant]
imgSub [filename]

Image Types

stdin	Float format
stdout	Float format
file	Float format

Description

This process performs a pixel-wise subtraction of images. Each input image is processed in turn and subtraction is performed in one of three ways: Either a constant provided on the command line is subtracted from all pixel values, or the image file named on the command line is subtracted in a pixel-wise manner, or else the next input image is subtracted in a pixel-wise manner. In the case of subtraction of two images, the image dimensions must be the same for both images.

Arguments

imgSub with no arguments subtracts image pairs from *stdin*. If a numeric is specified on the command line, this is interpreted as a constant to subtract. Otherwise, this argument is interpreted as the filename of an image to load and subtract.

Treatment of Valid Region

The valid region is not changed by this operator, but is ignored for the purposes of processing. The process is applied to the whole image.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are obtained during processing and are propagated with the output image.

See Also

imgAbs, imgAdd, imgMul, imgDiv, imgNeg.

imgSubSample

Name

imgSubSample—Perform sub-sampling on each image.

Syntax

imgSubSample [stride]
imgSubSample [x_stride y_stride]

Image Types

stdin	Any format
stdout	Any format

Description

This command sub-samples each input image leaving a smaller output image. The command takes samples at intervals specified on the command line, starting from the origin in the top left of the image.

Arguments

The stride parameters specify the x or y axis sampling intervals. If no parameters are present, the default stride is 2. A single parameter sets the stride in both dimensions equally.

Treatment of Valid Region

The valid region information is scaled down by the sampling interval.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCrop, imgMagnify, imgReorder.

imgSurfacePlot

Name

imgSurfacePlot—Surface viewing tool for X Windows.

Syntax

imgSurfacePlot [options...]

Image Types

stdin	Float image, Byte image (pgm)
stdout	Postscript

Description

imgSurfacePlot is a viewing tool that converts an image into a 3D surface to display under X Windows. The plot produced can be interactively positioned and then, when the desired view is achieved, the plot can be saved as a Postscript file. This tool can display data generated by the image processing tools described here or alternatively, from user programs. The program auto-scales the data so as to display it in the best manner. There are a number of command line options to control or override this feature. There is no limit to the number of times that a Postscript file can be saved from the same input data. Postscript output in portrait or landscape format can be printed or loaded into documents as required. For best effects, the input image size should be smaller than about 40 pixels square and the upper limit is 128 pixels square.

Command Line Options

-shade	Produce shaded plot in Postscript
-width *width*	Set startup window width (pixels)
-height *height*	Set startup window height (pixels)
-landscape	Produce "landscape" oriented Postscript
-paperwidth *width*	Set paper width, if not A4 (points)
-paperheight *height*	Set paper height, if not A4 (points)
-psfile *filename*	Set Postscript output file, else *stdout*
-scale *value*	Auto-scale magnification factor
-fixedscale *value*	Turn on fixed scaling with factor given
-pgm	Turn on fixed scaling for pgm images
-rotate *value*	Set initial viewpoint rotation
-elevate *value*	Set initial viewpoint elevation
-mesh	Draw as a mesh without hidden-line removal

Viewing Controls

Keyboard controls:	
p or Ctrl-p	Move viewpoint up by 10 degrees
n or Ctrl-n	Move viewpoint down by 10 degrees
f or Ctrl-f	Move viewpoint right by 10 degrees
b or Ctrl-b	Move viewpoint left by 10 degrees
P	Move viewpoint up by 1 degree
N	Move viewpoint down by 1 degree
F	Move viewpoint left by 1 degree
B	Move viewpoint right by 1 degree
Space	Print the current viewpoint angles
Return	Draw current view to Postscript output

Mouse controls:	
click right of centre	Move viewpoint right
click left of centre	Move viewpoint left
click above centre	Move viewpoint up
click below centre	Move viewpoint down

Input Format

Output from user programs can be viewed without difficulty as this tool can read a simple ASCII floating point image format. The header format is as follows: "ILibFloat width height 0 0 0 0 0 0". This is then followed by a list of width×height floats printed using the printf function, incrementing right and down through the array. Any of the white spaces can be carriage returns.

Treatment of Valid Region

The valid region is ignored by this tool.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are recomputed internally by this tool and do not therefore need to be set.

imgTee

Name

imgTee—Tee selected images from the stream to sequenced filenames.

Syntax

imgTee basename [start [incr [count [select_start [select_incr]]]]]

Image Types

stdin	Any format
stdout	Any format
file	Any format

Description

imgTee passes all images from its input to its output. In addition, selected images from the stream are written to sequenced filenames. The image files form a sequence with three digits in the filename indicating the sequence number. Files are written in ascending order starting from the file with sequence number *start* and incrementing by *incr* each time. At most *count* files are written. This command selects images from the stream for writing in the same manner as imgSelect.

Arguments

The filename argument specifies the basename of the file sequence. If a dot extension is specified, then the three digit sequence number will be inserted after the basename and before this extension. With no further command options, the default action is to write the file with the name exactly as given, without a sequence number. If the *start* parameter is given, this sets the sequence number of the first file. With no further options, imgTee writes files consecutively from this number until no further input is present. If an increment is specified then this is used to generate further sequence numbers. The parameter *incr* can be negative, but note that a file sequence number cannot be negative. The final parameter specifies the maximum number of files to write.

The *select_start* and *select_incr* parameters specify which of the incoming images are to be written to files. If *select_start* is set to zero, then writing will start from the first image. If the increment parameter is greater than one then some images are skipped between writing files. The default behaviour is to write alternate images starting from the first one, in the same manner as imgSelect.

Treatment of Valid Region

The valid region information is propagated unaltered by this function.

Treatment of Minimum and Maximum Values

The image minimum and maximum values are propagated unaltered by this function.

See Also

imgCombine, imgCombineSeq, imgInsert, imgRepeat, imgSelect,
imgSplit, imgSplitSeq.

imgThresh

Name

imgThresh—Threshold image.

Syntax

imgThresh [low_thresh [high_thresh]]

Image Types

stdin	Float format
stdout	Bit format (pbm)

Description

This command applies one or two thresholds to the input floating point image to yield a PBM image. The output value is set to white (0) if the input pixel value lies in the range $low\_thresh \leq v < high\_thresh$ otherwise it is set to black (1). The upper threshold may be missed out. The output mapping can me altered with the use of imgNot and associated Boolean operators.

Arguments

The low and high threshold values are supplied on the command line. If the high threshold is missing, no high limit is applied. If the low threshold is also missing, the image is thresholded at zero so that negative values are black (1) and the rest are white (0).

Treatment of Valid Region

Valid regions are not supported with this image type.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are not supported with this image type.

See Also

imgEdgeSynth, imgNot.

imgWrapAngle

Name

imgWrapAngle—Adjust angle data to lie in range $\pm\pi$.

Syntax

imgWrapAngle

Image Types

stdin	Float format
stdout	Float format

Description

This command adjusts values from the input image so that they lie in the range $\pm\pi$ allowing images containing angle information to be subtracted and added using the appropriate tools. The command imgWrapAngle acts as a postprocessor to correctly deal with angular wrap-around effects.

Treatment of Valid Region

The valid region is propagated unaltered by this function. All pixels are processed.

Treatment of Minimum and Maximum Values

Minimum and maximum values are calculated from the results of this process applied throughout the whole image.

See Also

imgAngle.

imgXor

Name

imgXor—Perform logical XOR between bit images.

Syntax

imgXor [filename]

Image Types

stdin	Bit format (pbm)
stdout	Bit format (pbm)
file	Bit format (pbm)

Description

This process performs a pixel-wise logical exclusive-OR of images. Each input image is processed in turn and the operation is performed in one of two ways: Either the image file named on the command line is XORed with the input image in a pixel-wise manner, or else the next input image is XORed in a pixel-wise manner. The image dimensions must be the same for both images.

Arguments

imgXor with no arguments operates on image pairs from *stdin*. Otherwise, the argument is interpreted as the filename of an image to load.

Treatment of Valid Region

Valid regions are not supported with this image type.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are not supported with this image type.

See Also

imgAnd, imgOr, imgNot.

imgZeros

Name

imgZeros—Zero-crossing detector.

Syntax

imgZeros [noborder]

Image Types

stdin	Float format
stdout	Bit format (pbm)

Description

imgZeros finds the positions of zero-crossings in each floating point input image. It forms part of a number of edge detection processes. The command uses a 3×3 kernel and detects zero-crossings in the following manner: Firstly, if the centre pixel is greater than zero and any of the vertically or horizontally surrounding pixels have negative values then this point is marked. Secondly, if the centre pixel happens to be zero, a test is made for "$+0-$" sequences in any direction: horizontally, vertically or diagonally. If any are found, the output point is marked. Zero-crossing segments appear black in the output image.

Arguments

The *noborder* argument if present turns off the border that is otherwise drawn around the perimeter of the valid region resulting from this operation. This argument can be anything.

Treatment of Valid Region

Valid regions are not supported with this image type.

Treatment of Minimum and Maximum Values

Minimum and maximum pixel values are not supported with this image type.

See Also

imgMarrHil, imgCanny, imgEdgeSynth.