**Leaf**

Basic image processing utility for Earth observation data

Will Grey

2016

**License**

This is free and unencumbered software released into the public domain. Anyone is free to copy, modify, publish, use, compile, sell, or distribute this software, either in source code form or as a compiled binary, for any purpose, commercial or non-commercial, and by any means.

In jurisdictions that recognize copyright laws, the author or authors of this software dedicate any and all copyright interest in the software to the public domain. We make this dedication for the benefit of the public at large and to the detriment of our heirs and successors. We intend this dedication to be an overt act of relinquishment in perpetuity of all present and future rights to this software under copyright law.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.

IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY CLAIM, DAMAGES OR

OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

For more information, please refer to <http://unlicense.org>

**Introduction**

Leaf is a basic image processing utility for Earth observation data. The tools are designed to be used in conjunction with other image processing systems, not instead of, although there is some overlap with some tools. As yet, Leaf does not have a means of visualising images, but the power of Leaf lies in its ability to be run from the command line thereby allowing automated and repetitive processing of large volumes of data. Leaf contains nearly 100 tools with others being added all the time. The native file format is flat file raw binary. This is free and unencumbered software released into the public domain. The tools are broadly categorised into one of nine groups:

* Image manipulation tools
* Statistical analysis tools
* Filter tools
* Text processing tools
* Classification tools
* Type conversion tools
* Format conversion tools
* Radar tools
* Miscellaneous tools

**List of tools**

**Image manipulation tools**

-xflip Flips images left to right

-yflip Flips images up to down

-swap Byte swapping from little endian to big endian and vice versa

-crop Extract sub area of an image

-rotate Rotate images

-resample Resamples raw binary image data using nearest-neighbour interpolation

-linear2Dinterp Perform 2D linear interpolation to upscale image

-shiftMeridian In global datasets shift meridian from middle to edge

-georeference Georeference Earth observation images

-mkimg Create a single value raw binary image file

-mosaic Mosaic images

-getPixel Extract pixel value from image

-transect Extract pixel values along a transect between two points

-diff Calculate the difference between two images

-add Add two images

-ratio Divide pixels values in image 1 by values in image 2

-regression Calculate regression between 2 images

-modulus Make all pixel values positive

-mask Apply mask to image

-thresh Apply threshold to image

-calibrate Apply gain and offset to image pixels

-rad2ref Convert from radiance values to reflectance

**Statistical analysis tools**

-covariance Calculate covariance and correlation matrix between channels

-stats Calculate univariate statistics of image channels

-hist Calculates histogram of multi channel input data file

-histoArea Calculates histogram of multi-channel input data file for each region

-anomalies Subtract image mean from pixel values

-standardise Subtract image mean from pixel values and divide by standard deviation

-addNoiseImg Box Muller transform to add Gaussian noise to image

-areaCounter Count pixels in 1/0 binary value byte image

-meanArea Calculates the statistics with regions

-mean Calculate mean of sequence of input images

-stdev Calculate standard deviation of sequence of input images

**Filter tools**

-modeFilter Apply mode (majority) filter to byte image data

-meanFilter Apply mean filter

-texture Apply texture filter

**Text processing tools**

-getNumLines Get number of lines in a file excluding blank lines

-getNumLinesAll Get number of lines in a file including blank lines

-calcTotalMean Calculate total and mean of a text file

-sideCat Concatonate 2 text files side by side

-rows2cols Converts rows to columns in text files

-addNoise Box Muller transform to add Gaussian noise to text file

-interpText Perform linear interpolation on column in a text file

**Classification tools**

-confusion Calculate confusion matrix for classified image

-classify Classify images using basic box classifier

-separability Calculate separability between classes for any number of channels

-reassign Reassign values within byte image

**Type conversion tools**

-byte2short Convert from signed char to signed short integer

-byte2float Convert from signed char to float

-byte2long Convert from signed char to signed long integer

-float2byte Convert from float to signed char

-long2float Convert from signed long integer to float

-long2byte Convert from signed long integer to signed char

-short2byte Convert from signed short integer to signed char

-short2float Convert from signed short integer to float

-short2long Convert from signed short integer to signed long integer

-long2short Convert from signed long integer to signed short integer

-ushort2float Convert from unsigned short integer to float

-ubyte2float Convert from unsigned char to float

-ubyte2ushort Convert from unsigned char to unsigned short integer

-ushort2ubyte Convert from unsigned short integer to unsigned char

-float2ubyte Convert from float to unsigned char

-long2ulong Convert from long to unsigned long

-ulong2long Convert from unsigned long to long

-short2ushort Convert from short to unsigned short

-ushout2short Convert from unsigned short to short

-ubyte2byte Convert from unsigned char to char

-byte2ubyte Convert from char to unsigned char

**Format conversion tools**

-bsq2dimap Convert from band sequential image to dimap

-dimap2bsq Convert from dimap to band sequential image

-bip2bsq Convert from byte interleave by pixel to band sequence

-bil2bsq Convert from byte interleave by line to band sequence

-bil2bip Convert from byte interleave by line to byte interleave by pixel

-bip2bil Convert from byte interleave by pixel to byte interleave by line

-byte2text Convert from char to ASCII text

-short2text Convert from short int to ASCII text

-float2text Convert from float to ASCII text

-float2textRow Convert from float to ASCII text as row

-float2textSN Convert from float to ASCII text in scientific notation

-text2float Convert from ASCII text to float

-byte2pgm Converts from binary byte data to pgm

-byte2ppm Converts from binary byte data to ppm

-byte2ppm3 Converts from 3 channel byte data to ppm

-ppm2byte Converts ppm to binary byte data

-pgm2byte Converts pgm to binary byte data

**Radar tools**

-linear2dB Convert radar data from linear to dB values

-dB2linear Convert radar data from dB to linear values

-complex2RealImag Convert complex radar data to real and imaginary

-complex2PwrPhase Convert complex radar data to power and phase

**Miscellaneous tools**

-copy Copy binary data file

-header Create ENVI header file

-spectralResponse Calculate broad band spectral response given irriadiance and spectral response function

-demSlope Calculates the slope and aspect from an input DEM float image

-degConv Convert between decimal degrees and degrees, minutes and seconds and vice versa

**Radar Tools**

There are a limited set of tools for processing synthetic aperture radar data.

**Convert radar data from linear to dB values**

|  |
| --- |
| leaf -linear2dB inImg outImg [null value]  inImg input image (float)  outImg output image (float)  null value default: 0.0 |

**Convert radar data from dB to linear values**

|  |
| --- |
| leaf -dB2linear inImg outImg [null value]  inImg input image (float)  outImg output image (float)  null value default: 0.0 |

**Convert complex radar data to real and imaginary**

|  |
| --- |
| leaf -complex2RealImag complexfile realfile imagfile  complexfile input complex image  realfile output real image  imagfile output imaginary image |

**Convert complex radar data to power and phase**

|  |
| --- |
| leaf -complex2PwrPhase complexfile realfile imagfile    complexfile input complex image  realfile output power image  imagfile output phase image |

**Type conversion tools**

There are a comprehensive set of tools for converting between different raw image file formats that are listed above. The syntax for the use of one of the tools is given below. The syntax for all the others is just the same.

**Convert from signed char to signed short integer**

|  |
| --- |
| leaf -byte2short infile [outfile]    infile Input image  outfile Output image |

The square brackets denote an optional argument. If the argument is not specified the output image name is the same as the input name with the *.out* suffix.

**Filter tools**

**Apply mode (majority) filter to byte image data**

|  |
| --- |
| leaf -modeFlter inImg outImg xdim [winsize]    inImg Input image (byte)  outImg Output image (byte)  xdim Number of pixels per row  winsize Window size; odd only Default: 3 |

**Apply mean filter**

|  |
| --- |
| leaf -meanFlter inImg outImg xdim [winsize] [dataType]  inImg Input image (byte)  outImg Output image (byte)  xdim Number of pixels per row  winsize Window size; odd only Default: 3  dataType 1:byte (default); 2:short; 3:long; 4:float; 8: double |

**Apply texture filter**

|  |
| --- |
| leaf -texture inImg outImg xdim [winsize] [dataType]    inImg Input image (byte)  outImg Output image (byte)  xdim Number of pixels per row  winsize Window size; odd only Default: 3  dataType 1:byte (default); 2:short; 3:long; 4:float; 8: double |

**Classification tools**

**Calculate confusion matrix for classified image**

|  |
| --- |
| leaf -confusion groundTruthImage classifiedImage outputTextFile  groundTruthImage Input ground truth image  classifiedImage Input classified image  outputTextFile Output confusion matrix |

**Classify images using basic box classifier**

|  |
| --- |
| leaf -classify inImg outImg inTextFile channels classes  inImg Input image  outImg Output image (byte)  inTextFile Input class file  dataType 1:byte (default), 2:short, 3:long, 4:float, 8:Double  channels Number of channels  classes Number of classes |

**Calculate separability between classes for any number of channels**

|  |
| --- |
| leaf -separability inputtextfile <lines> <channels>  inputtextfile Text file of histograms (class i band  1,2,3....class j band 1,2,3  lines Number of lines  channels Number of channels |

**Reassign values within byte image**

|  |
| --- |
| Usage: leaf -reassign inImg inImg inputTextFile  inImg Input image (byte)  outImg Output image (byte)  inputTextFile Input two columns of ascii |

**Miscellaneous tools**

**Copy binary data file**

|  |
| --- |
| leaf -copy inImg [outImg]  inImg Input image  [outImg] Output image |

This is the same as the unix *cp* command.

**Create ENVI header file**

|  |
| --- |
| leaf -header filename xdim ydim channels dataType [byteOrder]  filename Name of header file to create  xdim Number of pixels in x  ydim Number of pixels in y  channels Number of channels  dataType 1: char, 2: short, 3: long, 4: float, 8: double  [byteOrder] 0: Little endian 1: Big endian |

The native file format of Leaf is the same as ENVI. Thus if header files are created then data are easily viewed in ENVI,

**Example use:**

|  |
| --- |
| $ leaf -header leafImg.hdr 100 200 10 2 0 |

**Creates a header file for an short integer image named *leafImg* that is 100 by 200 pixels in x and y dimensions respectively and contains 10 channels.**

**Calculate broad band spectral response given irriadiance and spectral response function**

|  |
| --- |
| leaf -spectralResponse srfFile [spectraFile/planck]  srfFile Spectral response function file  spectraFile For surface or atmospheric spectra / planck model |

**Calculates the slope and aspect from an input DEM float image**

|  |
| --- |
| leaf -demSlope inDEM outSlope outAspect xdim spacing  inDEM input DEM image  outSlope Output slope image  outAspect Output aspect image  xdim Number of pixels in x dimension  spacing Pixel spacing (m) |

**Convert between decimal degrees and degrees, minuts and seconds and vice versa**

|  |
| --- |
| leaf -degConv <degrees> [minutes] [seconds] |

**Example use:**

|  |
| --- |
| $ leaf -degConv -101 45 0  output: -101.75  $ leaf -degConv -95 45 0  output: -95.75  $ leaf -degConv -1 45 0  output: -1.75  $ leaf -degConv 54 45 0  output: 54.75    $ leaf -degConv 54.75  output: 54:45:0 |

**Text processing tools**

**Get number of lines in a file excluding blank lines**

|  |
| --- |
| leaf -getNumLines inputtextfile  inputtextfile Input text file |

**Get number of lines in a file including blank lines**

|  |
| --- |
| leaf -getNumLinesAll inputtextfile  inputtextfile Input text file |

**Calculate total and mean of a text file**

|  |
| --- |
| leaf -calcTotalMean inputtextfile  inputtextfile Input text file |

**Concatonate 2 text files side by side**

|  |
| --- |
| leaf -sideCat inputtextfile1 inputtextfile2 outputtextfile  inputtextfile1 Input text file 1  inputtextfile2 Input text file 2  outputtextfile Output text file |

**Converts rows to columns in text files**

|  |
| --- |
| leaf -rows2cols inImg outImg  inImg input text image  outImg output text image |

**Box Muller transform to add Gaussian noise to text file**

|  |
| --- |
| leaf -addNoise inputTextfile outputTextfile sigma  inputTextfile Input text file  outputTextfile Output text file  sigma Standard deviation of error |

**Perform linear interpolation on column in a text file**

|  |
| --- |
| leaf -interpText inFile outFile startVal interval endVal col skip  inFile Input text file  outFile Output text file  startVal Start Value  interval Interval  endVal End value  cols Number of columns to process  skip Skip lines at start of file |

**Image manipulation tools**

**Flips images left to right**

|  |
| --- |
| leaf -xflip inImg outImg xdims [bpp] [channels]  infile input image  outfile output image  xdim number of pixels per row  bpp bytes per pixel: 1,2,4, or 8  channels Number of channels |

**Flips images up to down**

|  |
| --- |
| leaf -yflip inImg outImg xdims [bpp] [channels]  infile input image  outfile output image  xdim number of pixels per row  bpp bytes per pixel: 1,2,4, or 8  channels Number of channels |

**Byte swapping from little endian to big endian and vice versa**

|  |
| --- |
| leaf -swap infile [outfile] [bpp]  infile Input image  outfile Output image  bpp bytes per pixel (2,4,8) |

**Extract sub area of an image**

|  |
| --- |
| leaf -crop inImg cropImg xdim xoffset yoffset xsize ysize bytesPerPixel [channels]  inImg Input image  outImg Output crop image  xdim Number of pixels per row  xoffset Start pixel in x  yoffset Start pixel in y  xsize Cropped image number of pixels in x  ysize Cropped image number of pixels in y  bytesPerPixel Bytes per pixel: 1,2,4, or 8  [channels] Number of channels |

**Rotate images**

|  |
| --- |
| leaf -rotate infile outfile xdim [angle] [bpp] [channels]  infile input image 1  outfile output image  angle angle of rotation  bpp Bytes per pixel  channels Number of channels |

**Resamples raw binary image data using nearest-neighbour interpolation**

|  |
| --- |
| leaf -resample inImg outImg xdim factor [channels] [bytesPerPixel]  inImg Input image  outImg Output image  xdim Number of pixel in x dimension  factor Resampling factor  channels Number of channels (default =1)  bpp Bytes Per Pixel (default =1) |

**Perform 2D linear interpolation to upscale image**

|  |
| --- |
| leaf -linear2Dinterp inImg outImg xdim xscale yscale [channels] [dataType]  inImg Input image  outImg Output image  xdim Number of pixels per row  xscale Scale in x  yscale Scale in y  [channels] Number of channels (default=1)  [dataType] Float Only |

**In global datasets shift meridian from middle to edge**

|  |
| --- |
| leaf -shiftMeridian inImg outImg xdims [bpp] [channels]  infile input image  outfile output image  xdim number of pixels per row  bpp bytes per pixel: 1,2,4, or 8  channels Number of channels |

**Georeference Earth observation images**

|  |
| --- |
| leaf -georeference infile outfile xdim gcpFile [bytesPerPixel]  infile Unregistered input image 1  outfile Georeferenced output image 2  xdim Number of pixels in x dimension  gcpFile Text file of GCP points  dataType 1 char; 2 short; 3 long; 4 float; double 8 |

**Create a single value raw binary image file**

|  |
| --- |
| leaf -mkimg outImg xdim ydim dataType pixVal [channels]  outImg Name of image to create  xdim Number of pixels in x dimension  ydim Number of pixels in y dimension  dataType 1:byte (default), 2:short, 3:long, 4:float,  8:Double  pixVal Pixel value  [channels] Number of channels |

**Mosaic images**

|  |
| --- |
| leaf -mosaic mosaicImg inImg mosaicXdim inXdim xoff yoff bytesPerPixel  mosaicImg Mosaic image to be overwritten  inImg Imput image  mosaicXdim Number of pixels per row in mosaic image  inXdim Number of pixels per row in input image  xoff Start pixel in x  yoff Start pixel in y  bytesPerPixel Bytes per pixel: 1,2,4, or 8 |

**Extract pixel value from image**

|  |
| --- |
| leaf -getPixelValue inImg xdim xpos ypos dataType [channels]  inImg Input image  xdim Number of pixels per row  xpos x position of pixel  ypos y position of pixel  dataType 1:byte; 2:short; 3:long; 4:float; 8:double  channels Number of channels (default=1) |

**Extract pixel values along a transect between two points**

|  |
| --- |
| leaf -transect inImg xdim xpos1 ypos1 xpos2 ypos2 [dataType] [Channels]  inImg Input image  xdim Number of pixels per row  xpos1 x position of pixel 1  ypos1 y position of pixel 1  xpos2 x position of pixel 2  ypos2 y position of pixel 2  [dataType] 1:byte; 2:short; 3:long; 4:float; 8:double  [channels] Number of channels |

**Calculate the difference between two images**

|  |
| --- |
| leaf -diff infile1 infile2 outfile [data\_type]  infile1 input image 1  infile2 input image 2  outfile output image  data\_type 1:byte (default), 2:short, 3:long, 4:float,  8:Double |

**Add two images**

|  |
| --- |
| leaf -add infile1 infile2 outfile [data\_type]  infile1 input image 1  infile2 input image 2  outfile output image  data\_type 1:byte (default), 2:short, 3:long, 4:float,  8:Double |

**Divide pixels values in image 1 by values in image 2**

|  |
| --- |
| leaf -ratio infile1 infile2 outfile [data\_type]  infile1 input image 1  infile2 input image 2  outfile output image  data\_type 1:byte (default), 2:short, 3:long, 4:float,  8:Double |

**Calculate regression between 2 images**

|  |
| --- |
| leaf -regression infile1 infile2 [outfile] [dataType]  infile1 input image 1  infile2 input image 2  outfile output image  data\_type 1:byte (default), 2:short, 3:long, 4:float,  8:Double |

**Make all pixel values positive**

|  |
| --- |
| leaf -modImg infile outfile [dataType]  infile input image 1  outfile output image  dataType 1:byte (default), 2:short, 3:long, 4:float,  8: Double |

**Apply mask to image**

|  |
| --- |
| leaf -mask infile maskfile (Byte) outfile [data\_type]  infile input image  mask file mask image  outfile output image  data\_type 1:byte (default), 2:short, 3:long, 4:float, 8:Double |

**Apply threshold to image**

|  |
| --- |
| leaf -thresh infile outfile max\_thresh min\_thresh [data\_type]  infile input image 1  outfile output image  threshold (max) upper threshold value  threshold (min) lower threshold value  data\_type 1:byte (default), 2:short, 3:long, 4:float,  8:Double |

**Apply gain and offset to image pixels**

|  |
| --- |
| leaf -calibrate inImg outImg gainOffsetFile [dataType] [channels] [ignoreZeroValue]  inImg input image  outImg output image  gainOffsetFile File containing gain and offset for each  channel:  outputVal = offset + gain \* inputVal  dataType 1:byte (default), 2:short, 3:long, 4:float,  8:Double  channels Number of channels (default=1)  ignoreZeroValue Ignore zero values (default=No(0)) |

**Convert from radiance values to reflectance**

|  |
| --- |
| leaf -rad2ref inImg outImg szaImg irradianceFile [dataType] [channels] [scale]  inImg input radiance image  szaImg input solar zenith angle image for each pixel  outImg output reflectance image  irradianceFile Text file of solar irrandiance values for each  band  dataType 1: byte (default), 2: short, 3: long, 4: float,  8: Double  channels Number of channels  scale Scale value if required default=1.0 |

**Statistical analysis tools**

**Calculate covariance and correlation matrix between channels**

|  |
| --- |
| leaf -covariance inImg [channels] [bytesPerPixel]  inImg Input image  channels Number of channels  dataType 1:byte (default), 2:short, 3:long, 4:float, 8:Double |

**Calculate univariate statistics of image channels**

|  |
| --- |
| leaf -stats inImg [channels] [dataType]  inImg Input image  channels Number of channels  dataType 1:byte (default), 2:short, 3: ong, 4:float, 8: Double |

**Calculates histogram of multi channel input data file**

|  |
| --- |
| leaf -hist inImg [channels] [dataType] [Nbins]  inImg Input image  channels Number of channels  dataType 1:byte (default), 2:short, 3:long, 4:float, 8: Double  Nbins Number of bins in histogram |

**Calculates histogram of multi-channel input data file for each region**

|  |
| --- |
| leaf -histoArea inImg inCls [channels] [bytesPerPixel]  inImg Input image  inCls Input classified image  channels Number of channels  dataType 1:byte (default), 2:short, 3:long, 4:float, 8: Double  Nbins Number of bins in histogram |

**Subtract image mean from pixel values**

|  |
| --- |
| leaf -anomalies inImg outImg [dataType] [channels] [IgnoreValue] [nullValue]  inImg input image  outImg output image (float)  dataType 1:byte (default), 2:short, 3:long, 4:float,  8: Double  channels 1 (default)  IgnoreValues 0: No (default), 1: yes  nullValue 0.0 (default) |

**Subtract image mean from pixel values and divide by standard deviation**

|  |
| --- |
| leaf -standardise inImg outImg [dataType] [channels] [IgnoreValue] [nullValue]  inImg input image  outImg output image (float)  dataType 1:byte (default), 2:short, 3:long, 4:float,  8: Double  channels 1 (default)  IgnoreValues 0: No (default), 1: yes  nullValue 0.0 (default) |

**Box Muller transform to add Gaussian noise to image**

|  |
| --- |
| leaf -addNoiseImg inImg outImg sigma [dataType]  inImg input image  outImg output image  sigma add standard deviation such that r = sigma \* z + mu  dataType 1:byte (default), 2:short, 3:long, 4:float, 8:Double |

**Count pixels in 1/0 binary value byte image**

|  |
| --- |
| leaf -areaCounter inImg  inImg input image (byte) |

**Calculates the statistics with regions**

|  |
| --- |
| leaf -meanArea inImg inCls [channels] [bytesPerPixel]  inImg Input image  inCls Input classified image  channels Number of channels  dataType 1:byte (default), 2:short, 3:long, 4:float, 8:Double |

**Calculate mean of sequence of input images**

|  |
| --- |
| leaf -mean infile outfile [dataType] [IgnoreValue] [nullValue]  infile input textfile of image list  outfile output image  dataType 1:byte (default), 2:short, 3:long, 4:float,  8:Double  IgnoreValues 0: No, 1: yes (default)  nullValue 0.0 |

**Calculate standard deviation of sequence of input images**

|  |
| --- |
| leaf -stdev infile outfile [dataType] [IgnoreValue] [nullValue]  infile input textfile of image list  outfile output image 2  dataType 1:byte (default), 2:short, 3:long, 4:float,  8:Double  IgnoreValues 0: No, 1: yes (default)  nullValue 0.0 |

**Format conversion tools**

**Convert from band sequential image to dimap**

|  |
| --- |
| leaf -bsq2dimap infile outdir [channels]  infile input image  outdir output directory  [channels] default = 3 |

**Convert from dimap to band sequential image**

|  |
| --- |
| leaf -dimap2bsq indir outfile  indir input path (Dimap folder) 1  outImg output image |

**Convert from byte interleave by pixel to band sequence**

|  |
| --- |
| leaf -bip2bsq infile outfile xdim [channels] [bytesPerPixel]  infile input image  outfile output image  xdim number of pixels per line  channels number of bands: default=3  bytesPerPixel default=1 (byte) |

**Convert from byte interleave by line to band sequence**

|  |
| --- |
| leaf -bil2bsq infile outfile xdim [channels] [bytesPerPixel]  infile input image  outfile output image  xdim number of pixels per line  channels number of bands: default=3  bytesPerPixel default=1 (byte) |

**Convert from byte interleave by line to byte interleave by pixel**

|  |
| --- |
| leaf -bil2bip infile outfile xdim [channels] [bytesPerPixel]  infile input image  outfile output image  xdim number of pixels per line  channels number of bands: default=3  bytesPerPixel default=1 (byte) |

**Convert from byte interleave by pixel to byte interleave by line**

|  |
| --- |
| leaf -bip2bil infile outfile xdim [channels] [bytesPerPixel]  infile input image  outfile output image  xdim number of pixels per line  channels number of bands: default=3  bytesPerPixel default=1 (byte) |

**Convert from char to ASCII text**

|  |
| --- |
| leaf -byte2text infile outfile xdim  infile Input image  outfile Output image  xdim Number of pixels in x dimension |

**Convert from short int to ASCII text**

|  |
| --- |
| leaf -short2text infile outfile xdim  infile Input image  outfile Output image  xdim Number of pixels in x dimension |

**Convert from float to ASCII text**

|  |
| --- |
| leaf -float2text infile outfile xdim  infile Input image  outfile Output image  xdim Number of pixels in x dimension |

**Convert from float to ASCII text as row**

|  |
| --- |
| leaf -float2textRow infile outfile xdim  infile Input image  outfile Output image  xdim Number of pixels in x dimension |

**Convert from float to ASCII text in scientific notation**

|  |
| --- |
| leaf -float2textSN infile outfile xdim  infile Input image  outfile Output image  xdim Number of pixels in x dimension |

**Convert from ASCII text to float**

|  |
| --- |
| leaf -text2float infile outfile  infile Input image  outfile Output image |

**Converts from binary byte data to pgm**

|  |
| --- |
| leaf -byte2pgm inImg outImg xdim nColours  inImg input byte image  outImg output pgm image  xdim number of pixels per row  nColours number of colours |

**Converts from binary byte data to ppm**

|  |
| --- |
| leaf -byte2ppm infile (byte) outfile (pgm) xdim  infile input byte image  outfile output pgm image  xdim number of pixels per row |

**Converts from 3 channel byte data to ppm**

|  |
| --- |
| leaf -byte2ppm3 infile (byte) outfile (pgm) xdim  infile input byte image  outfile output pgm image  xdim number of pixels per row |

**Converts ppm to binary byte data**

|  |
| --- |
| leaf -ppm2byte infile outfile  infile input ppm image  outfile output byte image |

**Converts pgm to binary byte data**

|  |
| --- |
| leaf -pgm2byte inImg outImg  inImg input pgm image  outImg output byte image |