Image Processing Toolbox for Julia

Release 1.0

Michael A. Wirth*
School of Computer Science
University of Guelph

January, 2017

Abstract

This note describes a toolbox of image processing algorithms for Julia.

Introduction

Julia is a new language for scientific computing, and due to its similarity to MATLAB, is an excellent conduit for image processing. This toolbox contains functions to perform elementary image processing operations. The toolbox is completely free of dependencies, meaning it does not require any other packages to work (except for the use of GadFly to visualize histograms).

The toolbox contains the following packages:
Image I/O: text image files, and PGM images
Binarization: Local and global thresholding algorithms
Colour spaces: Conversion from RGB to YIQ, HSV, YCbCr, and CIELab (and back)
Segmentation: General segmentation algorithms, e.g. histogram back projection
Edge processing: Edge enhancement and detection algorithms
Morphology: A vast repoitoire of functions for morphological analysis
Spatial transformation: Various geometrical algorithms, e.g. rotation, flipping
Image sharpening: various unsharp masking filters
Noise suppression: a series of varied filters to perform noise suppression
Histogram functions: generate and manipulate histograms, eg. histogram equalization
Noise generation: Functions to generate noise in images
Skin: Skin segmentation algorithms

Algorithms

Image Sharpening Filters

Traditional unsharp masking	`filter_sharpUSM`		`imageFILTER`
UM with Order Statistic Laplacian	`filter_sharpUMOSLap`		`imageFILTER`
UM with Laplacian of Gaussian	`filter_sharpUSMLofG`		`imageFILTER`
UM with Gaussian smoothing	`filter_sharpUSMgauss`		`imageFILTER`

Basic Filters

Image convolution	`filter_CONV`		`imageFILTER`

* Contact: mwirth@uoguelph.ca

Noise Suppression (smoothing)

Gaussian smoothing	`filter_GAUSSIAN`		`imageFILTER`
Median filtering	`filter_MEDIAN`		`imageFILTER`
Truncated median filter	`enh_truncMedian`		`imageENH`
Mean (averaging) filter	`filter_MEAN`		`imageENH`
Hybrid median filter	`enh_hybridMedian`		`imageENH`
Alpha-Trimmed Means filter	`enh_alphaTMean`		`imageENH`
Weighted-median filter	`filter_wMEDIAN`		`imageENH`

Histogram Functions

Generate image histogram	`getIMhist()`	gray	`imageHIST`
Calculate cumulative histogram	`cumulativeHist()`	gray	`imageHIST`
Histogram equalization	`histEQ()`	gray	`imageHIST`
Histogram hyperbolization	`histHYPER()`	gray	`imageHIST`
Bi-histogram equalization	`bihistEQ()`	gray	`imageHIST`

Colour Image Functions

Convert RGB to YIQ colour space	`rgb2yiq()`	colour	`colourSPACE`
Convert YIQ to RGB colour space	`yiq2rgb()`	colour	`colourSPACE`
Convert RGB to HSV colour space	`rgb2hsv()`	colour	`colourSPACE`
Convert HSV to RGB colour space	`hsv2rgb()`	colour	`colourSPACE`
Convert RGB to CIELab	`rgb2lab()`	colour	`colourSPACE`
Convert CIELab to RGB	`lab2rgb()`	colour	`colourSPACE`
Convert RGB to CIE XYZ	`rgb2xyz()`	colour	`colourSPACE`
Convert CIE XYZ to RGB	`xyz2rgb()`	colour	`colourSPACE`

Image Thresholding Functions (grayscale images)

C= clustering E = entropy S = shape
A = attribute S = spatial L = local

Binarize an image	`im2BW()`		`imageBINARIZE`
Calculate entropy	`entropy()`		`imageBINARIZE`
Otsu's algorithm	`otsu()`	C	`imageBINARIZE`
Minimum Error algorithm	`minimumError()`	C	`imageBINARIZE`
Maximum Entropy algorithm	`maximumEntropy()`	E	`imageBINARIZE`
Niblack algorithm	`niblack()`	L	`imageBINARIZE`
Bernsen's algorithm	`bernsen()`	L	`imageBINARIZE`

Sauvola's algorithm	`sauvola()`	L	`imageBINARIZE`
Moment preservation algorithm	`moments()`	A	`imageBINARIZE`
Histogram entropy algorithm	`entropyPun()`	E	`imageBINARIZE`

Image Input/Output

Input a grayscale text image	`imreadGray()`	gray	`imageIO`
Output a grayscale text image	`imwriteGray()`	gray	`imageIO`
Plot an image histogram	`plotIMGhist()`	gray	`imageIO`
Read a PGM image file header	`readPGMheader()`		`PGMimages`
Read a type P2/P5 PGM image	`readPGM()`	gray	`PGMimages`
Write a type P2/P5 PGM image	`writePGM()`	gray	`PGMimages`
Read a type P6 colour PGM image	`readPGMc()`	colour	`PGMimages`
Write a type P6 colour PGM image	`writePGMc()`	colour	`PGMimages`

Image Noise Generation Functions

Impulse noise	`impulse()`	gray	`imageNoise`
Gaussian noise	`gaussian()`	gray	`imageNoise`
Raleigh noise	`raleigh()`	gray	`imageNoise`
Negative exponential noise (speckle)	`speckle()`	gray	`imageNoise`
Gamma noise	`gamma()`	gray	`imageNoise`
Uniform noise	`uniform()`	gray	`imageNoise`

Spatial Transformation Functions

Flipping images vertically and horizontally	`flip()`	gray	`imageTRANS`
Rotate by 90 degree increments	`rotate()`	gray	`imageTRANS`
Rotation about an arbitrary degree	`rotatefree()`	gray	`imageTRANS`

Edge Algorithms

Sobel edge enhancement	`sobel(), sobel2()`	gray	`imageEDGE`
Prewitt edge enhancement	`prewitt()`	gray	`imageEDGE`
Kirsch edge enhancement	`kirsch()`	gray	`imageEDGE`
Canny edge detection	`canny()`	gray	`imageEDGE`

Segmentation Algorithms

Parametric segmentation	`parametricSEG()`	colour	`imageSEG`
Histogram backprojection	`backPROJECT()`	colour	`backPROJ`

Skin Segmentation Algorithms

Skin segmentation using RGB	`skinRGB()`	colour	`skinSeg`
Skin segmentation using RGB/HSV	`skinHSVrgb()`	colour	`skinSeg`

Morphological Analysis

Image union	`imUnion()`	gray	`imageSET`
Image intersection	`imIntersect()`	gray	`imageSET`
Image complement	`imComplement()`	binary	`imageSET`
Is an image binary?	`isBinary()`	binary	`imageSET`
Connected components labelling	`labelBW()`	binary	`imageSET`
Labelled region statistics	`regionStats()`	gray	`imageSET`
Remove region < size	`fillIt()`	gray	`imageSET`
Make a structuring element	`makeSE()`	binary	`imageMORPH`
Binary dilation	`dilate()`	binary	`imageMORPH`
Binary erosion	`erode()`	binary	`imageMORPH`
Binary opening	`open()`	binary	`imageMORPH`
Binary closing	`close()`	binary	`imageMORPH`
Grayscale dilation	`dilateG()`	gray	`imageMORPH`
Grayscale erosion	`erodeG()`	gray	`imageMORPH`
Grayscale opening	`openG()`	gray	`imageMORPH`
Grayscale closing	`closeG()`	gray	`imageMORPH`
Morphological sharpening	`sharpen()`	gray	`imageMORPH`
Toggle contrast enhancement	`toggleCE()`	gray	`imageMORPH`
Top-hat filtering	`tophat()`	gray	`imageMORPH`
Bottom-hat filtering	`bothat()`	gray	`imageMORPH`
Morphological contrast enhancement	`morphCE()`	gray	`imageMORPH`
Alternating sequential filter	`morphASF()`	gray	`imageMORPH`
Conditional dilation	`condDilate()`	gray	`imageMORPH`
Area opening	`areaOpen()`	gray, binary	`imageMORPH`