Switching over to SimpleCV.

Copyright ©2012 SimpleCV.

June 4, 2012

SimpleCV[1], which stands for Simple Computer Vision, is an easy-to-use Python frame-work that bundles together open source computer vision libraries and algorithms for solving problems. The idea of this document is to provide a quick reference for switching from Matlab[2] and OpenCV[3]to SimpleCV.

Description	Matlab	OpenCV	SimpleCV
Reading an image	imread('lenna.png')	cvLoadImage('lenna.png')	Image('lenna.png')
Converting the image to RGB colorspace	hsv2rgb(hsv_image) or ind2rgb(X, map)	CvtColor(bitmap, retVal, CV_BGR2RGB)	img.toRGB()
Converting the image to BGR colorspace	-	CvtColor(bitmap, retVal, CV_RGB2BGR)	img.toBGR()
Converting the image to HLS colorspace	-	CvtColor(bitmap, retVal, CV_RGB2HLS)	img.toHLS()
Converting the image to HSV colorspace	rgb2hsv(rgb_image)	CvtColor(bitmap, retVal, CV_RGB2HSV)	img.toHSV()

[1]References : O'Reilly Publication, Practical Computer Vision with SimpleCV by Nathan Oostendorp, Anthony Oliver, and Katherine Scott.
[2]Matlab documentation : `http://www.mathworks.in/help/techdoc/`
[3]OpenCV documentation : `http://docs.opencv.org/`

Description	Matlab	OpenCV	SimpleCV
Converting the image to XYZ colorspace	cform = makecform('srgb2xyz'); applycform(rgb,cform);	CvtColor(bitmap, retVal, CV_RGB2XYZ)	img.toXYZ()
Converting the image to GRAY colorspace	rgb2gray(rgb_image)	CvtColor(bitmap, retVal, CV_RGB2GRAY)	img.toGray()
Create a new, empty OpenCV bitmap	zeros(H, W, C)	SetZero(bitmap)	img.getEmpty(channels)
Full copy of the image	newimg = img	Copy(bitmap, newimg)	img.copy()
Resize the image	imresize(img, scale)	Resize(bitmap, scaled_bitmap)	img.resize(x,y)
Invert image	imcomplement(img)	-	img.invert()
Horizontally mirror an image	flipdim(img,2)	Flip(bitmap, newimg_bitmap, 1)	img.flipHorizontal()
Vertically mirror an image	flipdim(img,1)	Flip(bitmap, newimg_bitmap, 0)	img.flipVertical()
Stretch filter on a greyscale image	img(img<th_l) = 0; img(img>th_h) = 255	Threshold(grayscale_bitmap, newimg, thresh_low, 255,CV_THRESH_TOZERO)	img.stretch(thresh_low, thresh_high)
Binary threshold of the image	- step(vision.Autothresholder,img)	Threshold(bitmap, bitmap, thresh, maxv, CV_THRESH_BINARY_INV)	img.binarize(thresh, maxv, blocksize, p)
Mean color of the image	mean(reshape(im, size(im,1)*size(im,2), size(im,3)))	Avg(bitmap)[0:3]	img.meanColor()
Finds the FeatureSet strongest corners first	corner(img)	GoodFeaturesToTrack(GrayscaleBitmap, eig_image, temp_image, maxnum, minquality, mindistance, None)	img.findCorners(maxnum, minquality, mindistance)

Description	Matlab	OpenCV	SimpleCV
Blobs are continuous light regions	step(vision.BlobAnalysis, fg_img)	-	img.findBlobs(threshval, minsize, maxsize, threshblocksize, threshconstant)
Finding the location of a known object	-	HaarDetectObjects(EqualizedGrayscaleBitmap(), cascade.getCascade(), storage, scale_factor, use_canny)	findHaarFeatures(self, cascade, scale_factor, min_neighbors, use_canny)
Uploading the Image to Imgur or Flickr	-	-	img.upload(dest,api_key, api_secret,verbose)
Smooth the image	H = fspecial(*type*); imfilter(I,H)	Smooth(r, ro, algorithm, win_x, win_y, sigma, spatial_sigma)	img.smooth(algorithm_name, aperature, sigma, spatial_sigma, grayscale)
Draw a circle on the Image	step(vision.MarkerInserter, img, pts)	-	img.drawCircle(ctr, rad, color, thickness)
Draw a line	plot(X_vector, Y_vector)	-	img.drawLine(pt1, pt2, color, thickness)
Size of image	[size(img,1) size(img,2)]	GetSize(bitmap)	img.size()
Split the image into a series of image chunks	-	-	img.split(cols, rows)
Images of R,G,B channels are recombined into a single image	cat(3, r, g, b)	Merge(b,g,r,None,retVal)	img.mergeChannels(r,b,g)
Apply a color correction curve in HSL space	-	-	img.applyHLSCurve(hCurve, lCurve, sCurve)
Apply a color correction curve in RGB space	-	-	img.applyRGBCurve(rCurve, gCurve, bCurve)
Applies Intensity to all three color channels	-	-	img.applyIntensityCurve(curve)
Returns Image of the string	-	-	img.toString()

Description	Matlab	OpenCV	SimpleCV
Split the channels of an image into RGB	r=img(:,:,1); g=img(:,:,2); b=img(:,:,3)	Split(bitmap, b, g, r, None)	img.splitChannels(grayscale)
Returns image representing the distance of each pixel from a given color tuple	-	-	img.colorDistance(color)
Apply morphological erosion to a image	imerode(img,SE)	Erode(bitmap, retVal, kern, iterations)	img.erode(iterations)
Apply morphological dilation to a image	imdilate(img,SE)	Dilate(bitmap, retVal, kern, iterations)	img.dilate(iterations)
Histogram equalization on the image	histeq(img, hgram)	cv.EqualizeHist(GrayscaleBitmap, Equalizedgraybitmap)	img.equalize()
Applies erosion operation followed by a morphological dilation	imerode(img, SE)	MorphologyEx(bitmap, retVal, temp, kern, CV_MOP_OPEN, 1)	img.morphOpen()
The difference between the morphological dilation and the morphological gradient	-	MorphologyEx(Bitmap, retVal, temp, kern, CV_MOP_GRADIENT, 1)	img.morphGradient()
1D histogram(numpy array) of intensity for pixels in the image	step(vision.Histogram,img)	-	img.histogram(numbins)
The histogram of the hue channel for the image	-	-	img.hueHistogram(bins)
Returns the peak hue values histogram of hues	-	-	img.huePeaks(bins)
Add two images	imadd(img1,img2)	Add(imgBitmap, otherBitmap, newBitmap)	img.__add__(other)

Description	Matlab	OpenCV	SimpleCV
Subtract two images	imsubtract(img1,img2)	Sub(imgBitmap, otherBitmap, newBitmap)	img.__sub__(other)
Or two images	-	Or(imgBitmap, otherBitmap, newBitmap)	img.__or__(other)
Image division operation taking two images as input	imdivide(img1,img2)	Div(imgBitmap, otherBitmap, newBitmap)	img.__div__(other)
Raises every array element in image array to a power	img.^p	Pow(imgBitmap, otherBitmap, other)	img.__pow__(other)
Finds 2d and 1d barcodes in the image	-	-	img.findBarcode(zxing_path)
Finds line segments in the image	hough(BW)	HoughLines2(em, CreateMemStorage(), CV_HOUGH_PROBABILISTIC, 1.0, CV_PI/180.0, threshold, minlinelength, maxlinegap)	img.findLines(threshold, minlinelength, maxlinegap, cannyth1, cannyth2)
Finds a chessboard within that image	-	FindChessboardCorners(EqualizedGrayscaleBitmap, dimensions, CV_CALIB_CB_ADAPTIVE_THRESH + CV_CALIB_CB_NORMALIZE_IMAGE)	img.findChessboard(dimensions, subpixel)
Canny edge detection method	edge(img,'canny')	Canny(GrayscaleBitmap, edgeMap, t1, t2)	img.edges(t1, t2)
function rotates an image around a specific point by the given angle	imrotate(img,angle)	GetRotationMatrix2D(point , angle, scale, rotMat)	img.rotate(angle, fixed, point, scale)
return a shear-ed image from the cornerpoints	tform = maketform('affine',A); imtransform(img,tform)	GetAffineTransform(src, cornerpoints, aWarp)	img.shear(cornerpoints)
Function for warp performs an affine rotation	tform = maketform('projective',A); imtransform(img,tform)	cv.WarpPerspective(imgBitmap, retVal, rotMatrix)	img.transformPerspective(rotMatrix)
Returns the RGB value for a particular image pixel	img(y, x, :)	Get2D(Bitmap, y, x)	img.getPixel(x, y)

Description	Matlab	OpenCV	SimpleCV
Returns a single row of RGB values from the image	squeeze(img(row, :, :))	GetRow(imgBitmap, row)	img.getHorzScanline(row)
Returns a single column of gray values from the image	gray=rgb2gray(img); squeeze(gray(:, column, :))	GetCol(imgGrayscaleBitmap, column)	getVertScanlineGray(column)
Returns a single row of gray values from the image	gray=rgb2gray(img); squeeze(gray(row, :, :))	GetRow(imgGrayscaleBitmap, row)	getHorzScanlineGray(row)
Crops the image based on parameters	imcrop(img, rect)	-	img.crop(x , y, w, h, centered)
Returns the selected region.	imrect(hparent,position)	-	img.regionSelect(x1, y1, x2, y2)
Clears out the entire image	img(:)=0	SetZero(Bitmap)	img.clear()
Draws the string on the image at the specified coordinates.	text(x,y,'string')	-	img.drawText(text , x , y , color, fontsize)
Draw a rectangle on the image	rectangle('Position',[x,y,w,h])	-	img.drawRectangle(x,y,w,h,color,width,alpha)
Shows the current image	imshow(img)	ShowImage("Image", image)	img.show(type)
Push a new drawing layer onto the back of the layer stack	-	-	img.addDrawingLayer(layer)
Insert a new layer into the layer stack at the specified index	-	-	img.insertDrawingLayer(layer, index)
Remove a layer from the layer stack based on the layer's index	-	-	img.removeDrawingLayer(index)
Return a drawing layer based on the index	-	-	img.getDrawingLayer(index)

Description	Matlab	OpenCV	SimpleCV
Returns the gray value for a particular image pixel	gray=rgb2gray(img); gray(y,x)	Get2D(GrayscaleBitmap(), y, x)	img.getGrayPixel(x, y)
Returns a single column of RGB values from the image	squeeze(img(:, column, :))	GetCol(imgBitmap, column)	img.getVertScanline(column)
Remove all of the drawing layers	-	-	img.clearLayers()
Return the array of DrawingLayer objects	-	-	img.layers()
Return all DrawingLayer objects as a single DrawingLayer.	-	-	img.mergedLayers()
Render all of the layers onto the current image	-	-	img.applyLayers(indicies)
automatically adjust image size to match the display size	imshow(img,'InitialMagnification','fit')	-	img.adaptiveScale(resolution,fit=True)
Combine two images as a side by side images	-	-	img1.sideBySide(img2, side, scale)
Generate a binary mask of the image based on a range of rgb values	[X,map] = rgb2ind(img, 65536); roicolor(X,low,high)	-	createBinaryMask(self,color1,color2)
Make the canvas larger but keep the image the same size	-	-	img.embiggen(size, color, pos)
The white areas of the mask will be kept and the black areas removed	X2 = zeros(size(X), 'uint16'); X2(mask) = X(mask); ind2rgb(X2, map)	-	img.applyBinaryMask(mask,bg_color)

Description	Matlab	OpenCV	SimpleCV
Generate a grayscale or binary mask image based either on a hue or an RGB triplet	-	-	img.createAlphaMask(hue, hue_lb,hue_ub)
Apply a function to every pixel and return the result	-	-	img.applyPixelFunction(theFunc)
Calculate the integral image and return it as a numpy array	integralImage(img)	Integral(GrayscaleBitmap,img)	img.integralImage(tilted)
Convolution performs a shape change on an image.	conv2(img,kernel,'shape')	Filter2D(Bitmap,retVal,myKernel,center)	img.convolve(,kernel,center)
Function searches an image for a template image	step(vision.TemplateMatcher, rgb2gray(img),rgb2gray(T))	-	img.findTemplate(template_image, threshold, method)
Return any text it can find using OCR on the image	-	-	img.readText()
extract perfect circles from the image	-	-	img.findCircle(canny,thresh,distance)
Attempts to perform automatic white balancing	-	-	img.whiteBalance(method)
Apply a LUT (look up table) to the pixels in a image	intlut(A, LUT)	LUT(bitmap,bitmap,fromarray(LUT))	img.applyLUT(rLUT,bLUT,gLUT)
Finds keypoints in an image and returns them as the raw keypoints	detectSURFFeatures(img)	-	img._getRawKeypoints(thresh,flavor, highQuality, forceReset)
Method does a fast local approximate nearest neighbors (FLANN) calculation between two sets of feature vectors	matchFeatures(feat1,feat2)	-	img._getFLANNMatches(sd,td)

Description	Matlab	OpenCV	SimpleCV
Calculates keypoints for both images, determines the keypoint correspondences	-	-	img.drawKeypointMatches(template, thresh, minDist,width)
Match a template image with another image using SURF keypoints.	-	-	img.findKeypointMatch(template, quality,minDist,minMatch)
This method finds keypoints in an image and returns them as a feature set	detectSURFFeatures(img)	-	img.findKeypoints(min_quality, flavor,highQuality)
Returns the colors in the palette of the image	-	-	img.getPalette(bins,hue)
Takes in the palette from another image and attempts to apply it to this image	-	-	img.rePalette(palette,hue)
returns the visual representation (swatches) of the palette in an image	-	-	img.drawPaletteColors(size,horizontal,bins,hue)
The method then goes through and replaces each pixel with the centroid of the clutsters found by k-means	-	-	img.palettize(bins,hue)
Palettization and behaves similar to the fndBlobs	-	-	img.findBlobsFromPalette(palette_selection, minsize, maxsize)
Method uses the color palette to generate a binary image	-	-	img.binarizeFromPalette(palette_selection)
Returns the RAW DFT transform of an image	fft2(X)	DFT(src, dst,CV_DXT_FORWARD)	img.rawDFTImage(grayscale)
Method applies a simple band pass DFT filter	-	-	img.bandPassFilter(xCutoffLow, xCutoffHigh, yCutoffLow, yCutoffHigh,grayscale)

Description	Matlab	OpenCV	SimpleCV
Skeletonization of the image	bwmorph(BW,'skel')	-	img.skeletonize(radius)
smartThreshold uses a method graph cut, to automagically generate a grayscale mask image	-	grabCut(npimg,mask,rect,tmp1,tmp2,10,mode)	img.smartThreshold(mask, rect)
It takes a image converts it to grayscale, and applies a threshold	-	-	img.smartFindBlobs(mask,rect,thresh_level)
This method is same as Paint bucket tool in image manipulation program	imfill(BW,locations)	FloodFill(bmp,tuple(points),color, lower,upper,flags)	img.floodFill(points,tolerance,color, lower,upper,fixed_range)
Returns Image where the values similar to the seed pixel have been replaced by the input color	-	-	img.floodFillToMask(points,tolerance, color,lower,upper,fixed_range,mask)
A featureset of blobs form the Mask Image	-	-	img.findBlobsFromMask(mask,threshold, minsize, maxsize)
Returns the log value of the magnitude image of the DFT transform	-	-	img.getDFTLogMagnitude(grayscale)
Apply an arbitrary filter to the DFT of an image	-	-	img.applyDFTFilter(flt,grayscale)
Applies a high pass DFT filter	-	-	img.highPassFilter(xCutoff,yCutoff,grayscale)
Applies a low pass DFT filter	-	-	img.lowPassFilter(xCutoff,yCutoff,grayscale)
Method performs an inverse discrete Fourier transform	ifft2(X)	-	InverseDFT(raw_dft_image)
DFT is applied on image using gaussian lowpass filter	-	-	img.applyUnsharpMask(boost,dia,grayscale)

Description	Matlab	OpenCV	SimpleCV
Performs an optical flow calculation and attempts to find motion between two subsequent frames	step(vision.OpticalFlow,img1,img2)	CalcOpticalFlowHS(previousFrameGrayscaleBitmap, imgGrayscaleBitmap,block,shift,spread,0,xf,yf)	img.findMotion(previous_frame, window, method, aggregate)
Creates a butterworth filter of 64x64 pixels, resizes it to fit the image	-	-	img.applyButterworthFilter(dia, order,highpass,grayscale)
Creates a gaussian filter of 64x64 pixels, resizes it to fit image	H = fspecial('gaussian',hsize,sigma); imfilter(I,H)	-	img.applyGaussianFilter(dia, highpass, grayscale)