Examples of Matlab Image Processing Toolkit

Professor Carl Grant Looney, Fall 2005

```
>> whos                  //no semicolon, shows info. on all images in memory
>> clear, close all;     //clears images from screen, closes all data in memory

>> Im1 = imread(‘lena256.tif’);         //read in an image
>> imshow(Im1);                         //show image on screen in Matlab frame
>> figure, imshow(I2), title(‘Fig. 2. Original image.”); //puts title on shown image

>> imwrite(Im2,’lenaout.gif’);          //writes Im2 as file of name and format given
>> iminfo(‘lenaout.gif’)                //no semicolon, shows info. about image

>> Imbkgrnd = imopen(I2,strel(‘disk’,20); //erodes, dilates light object pixels, forms background
>> I3 = imsubtract(I2,Imbkgrnd);        //subtracts background
>> I4 = imadjust(I3,stretchlim(I3),[0 1]); //stretches contrast of I3

>> myse = strel(‘rectangle’,30,40);     //creates structure elelment 30x40 rect. of 1's
>> Ibw2 = imerode(Ibw1,myse);           //erodes Ibw1 using structure element myse
>> Ibw3 = imdilate(Ibw1,myse);          //dilates Ibw1 using structure element myse
>> Ibw = im2bw(Im0,graythresh(I0)); //converts to b&w using computed threshold

>> Ibw1 = edge(I1, ‘sobel’);            //edge detection via sobel method
>> Ibw2 = edge(I1,’canny’);             //edge detection via canny method
>> I2 = histeq(I1);                     //does histogram equalization on I1

>> I2 = imresize(I1,1.5);               //makes image 1.5x1.5 times larger
>> I2 = imresize(I1,1.2,’bicubic’);     //uses bicubic interpolation
>> I2 = imresize(I1,1.2,’bilinear’);    //uses bilinear interpolation
>> I2 = imresize(I1,1l4,’nearest’);     //uses nearest neighbor interpolation

>> mytransform = maketform(‘affine’,[1.2 1.1 0; 0.66 1.2 0; 0 0 1]; //makes affine transf.
>> Im3 = imtransform(Im2,mytransform); //transforms Im2 by matrix (affine transf.)

>> I0 = imcomplement(I2);               //takes complement (inversion of I2)
>> I2 = imabsdiff(I0,I1);               //absolute difference of images
>> I3 = imadd(I1,I2);
>> I4 = imdivide(I1,I2);
>> I5 = immultiply(I1,I2);
>> I6 = immultiply(I1,1.4);
>> I7 = imsubtract(I1,I2);
>> I8 = imadd(I1,40);
>> I9 = imsubtract(I1,20);
>> I10 = imlincomb(0.4,I1,0.6,I2);     //I10 is linear combination 0.4* I1 + 0.6*I2 pixelwise
>> I3 = double(Ibw1) + double(Ibw2); //converts bw to double and adds

>> I11 = filter2([1 2; -1 2], I1);         //filters I1 by array given (usually a mask filter)
>> I12 = imrotate(I1,35,’bilinear’);    //rotate 35° and use bilinear interpolation
>> I13 = imcrop;                                   //must click upper left, drag to lower right to get I13
                                                             //from current focused image
>> I2 = imcrop(I1);                              //permits cropping of specified image I1
>>X2 = imcrop(X,map);                      //X is indexed image with color map

>> h = fspecial(‘unsharp’);                //get unsharp masking mask
>> I14 = imfilter(I1,h);                       //unsharp masking filter with h from above
>> I15 = imfilter(I1,h,’replicate’);      //replicates border pixels so mask fits border pixels

>> Ibw2 = bwmorph(Ibw1,’dilate’);  //dilate Ibw1
>> Ibw3 = bwmorph(Ibw1,’erode’);   //erode Ibw1
>> Ibwout = Ibw2 - Ibw3;                 //subtract eroded from dilated, get thick boundaries
>> Ibwthin = bwmorph(Ibwout,’thin’); //thin the resulting boundaries

>> myse = strel(‘diamond’,5);            //make diamond shaped block (structuring element
>> myse = strel(‘disk’,10);
>> myse = strel(‘square’,8);
>> myse = ones(5);                             //make a 5x5 array of all “1's”
>> bw2 = imdilate(bw1,myse);           //use 5x5 array as structuring element in dilation
>> myse = ones(15,20);                       //15 by 20 array of “1's”
>> bw3 = imerode(bw1,myse);           //erode image with 15 by 20 array

>>Im1 = imread(‘pollens.tif’);             //read in pollens image
>> Im2 = imcomplement(imfill(imcomplement(Im1), holes)) //fills in light gray holes in
                                                             // dark objects - works on complements
>> Im3 = imnoise(Im1,’gaussian’,0,0.005); //put Gaussian noise on 0.5% of pixels
>> Im4 = imnoise(Im1,’salt & pepper’, 0.02); put on 2% salt & pepper noise
>> Im5 = medfilt2(Im4, [3 3]);           //use 3x3 median filter to take off noise

>> mean2(Im1)                                   //hit enter with no semicolon to get mean value
>> std2(Im1)                                       //hit enter (no semicolon) to get std. deviation
>> gamma = 0.8;                                //gamma > 1 => darker, gamma < 1 => lighter
>> Im2 = imadjust(I1,[],[],0.5);          //adjusts gamma: exponential map of grays

>> I2 = nlfilter(I1, [3 3], ‘std2');          //nonlinear fltr: 3x3 nbhd std. deviation is output new pixel
>> I2 = nlfilter(I1, [5 5], @myfun);    //nonlinear fltr: 5x5 nbhd pixels are inputs to defined
                                                                  function myfun that outputs new pixel value
```

```
>> I2 = nlfilter(I1,[3 4], ‘max(x(:))’); //new pixel is max. of 3x4 nbhd pixels in I1
>> I2 = nlfilter(I1,[3 3], ‘mean2');      //new pixel is mean of 3x3 nbhd pixels in I1

>> I1 = imread(‘pollens.gif’);            //read in pollens image
>> imshow(I1);                            //show image on screen
>> Ibw = roipoly;                         //use clicks at points on image to connect as polygon
>> figure, imshow(Ibw);                   //shows interior of polygon, remainder is blacked out
>> I2 = immultiply(I1,Ibw);               //multiplies original by Ibw, blacks out all but polygon part
>> figure, imshow(I2);                    //shows only interior of polygon in original grays, so this
                                          //can be added to another image, or used some way

>> Abw1 = zeros(9,10);                    //creates bw array of 9x10 “0's”
>> Abw1 = (4:5, 4:7) = 1;                 //changes 0 to 1 in rows 4 to 5, for cols. 4 to 7

>> bw1 = zeros(20,20);                    //creates 20x20 block of zeros
>>bw1(4:6,6:10) = 1;                      //puts 1's in rows 4 to 6 under columns 4 to 7

>> h = ones(5,5)/25;                      //5x5 block of values 1/25
>> I2 = imfilter(I1,h);                   //filter I1 with h

>> [X1,map1] = imread(‘forestscene.tif’); //reads image for placing
>> [X2,map2] = imread(‘lake.tif’);        //reads image for placing
>> subplot(1,2,1), imshow(X1,map1); //1 row,2 cols; places in row 1, column 1
>> subplot(1,2,2), imshow(X2,map2); //1 row, 2 cols; places in row 1, column 2

>> imshow myFlowers.tif                   //no semicolon, shows image for selecting line
>> improfile                              //now draw line (click point and drag) on image and either
                                          //intensity, or 3 R, G, B profiles graphs are drawn
>> figure, imcontour(I1)                  //draws contours around light objects

>> I4ims = cat(4,I1,I2,I3,I4);            //concatenates the 4 frames (images of same type &size
>> Ifrm3 = I4ims(:,:,:,3);                //extracts frame I3 from I4ims, puts it in Ifrm3
>> Imovie = immovie(I4ims,map);           //creates a movie of the frames in I4ims and makes a color
                                          //map, but frames must be indexed image type
>> movie(Imovie);                         //plays the movie of all frames in I4ims, but only if
                                          //frames are indexed images with a color map
```