Lab 2: Introduction to Image Processing

(Written in part by Dr. A. Miguel of Seattle University)

Note: Each group provides only one report for all group members. All MATLAB files (.m files) must be uploaded.

1. Purpose

The purpose of this lab is to introduce you to some basic concepts in image processing. You will learn how to read and display an image in MATLAB. You will perform simple edge detection on an image we give you as well as on an image of your own. Then, you will scale an image to create its thumbnail version. If you enjoy this lab, make sure to take the following higher-level undergraduate classes related to image processing that are offered by the EE and CSE departments: EE 440, CSE 455, and CSE 457.

2. Background

Digital images consist of pixels (picture elements). When the pixels are placed close to each other, an image viewed on a computer display or printed on a paper appears to be continuous. The number of pixels per inch (ppi) varies with an application. Some monitors can only display 72 ppi. For publishing, 200–1200 ppi is often required. Laser printers are usually capable of 300-600 ppi. The brightness and color information for each pixel is represented by a number in a two-dimensional array (matrix). The location in the matrix corresponds to the location of a pixel in the image. For example, $X[1,1]$ (usually) identifies the pixel located in the upper left corner, as shown in figure 1. The pixel values in an 8-bit gray scale image can take any value from 0 to 255. Usually black is encoded by a value of 0 and white by a value of 255. A color image is stored in a three-dimensional array, where the first plane in the 3rd dimension represents the red pixel intensities, the second plane represents the green pixel intensities, and the third plane represents the blue pixel intensities. True color has 24 bits of resolution (8 bits for each of the red, green, and blue planes).

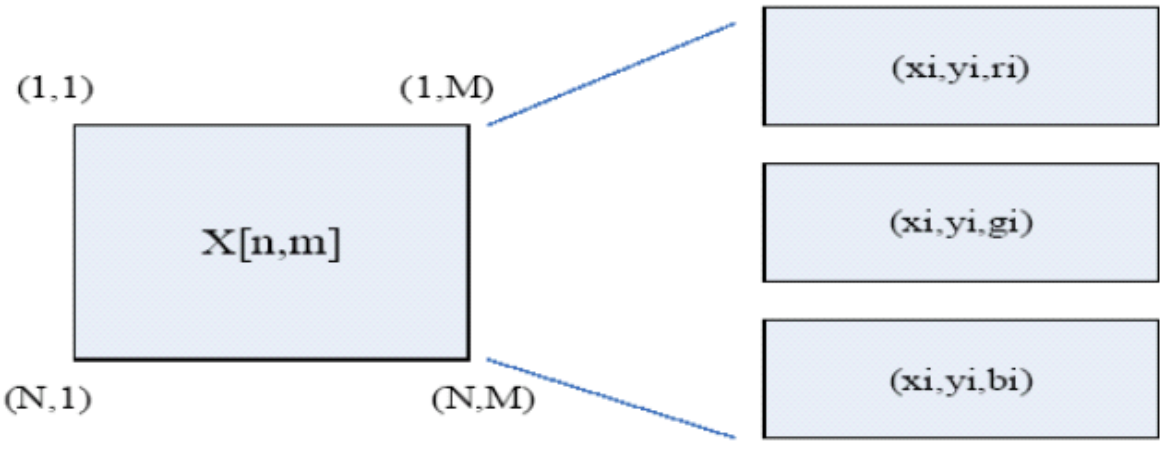


Figure 1. Pixel locations and file format; ri=r_i represents red pixel intensity at location (xi,yi) = (x_i,y_i) while gi=g_i and bi=b_i are green and blue pixel intensities, respectively.

In this lab, you will use the MATLAB Image Processing Toolbox. To learn more about the functions included in it, type help images.

Assignment 1

Go to the EE341 web page and download the file DailyShow.jpg. Since your image is in the JPEG format, you will need to tell this to Matlab when you read it in. Use the MATLAB command:

imread('DailyShow', 'jpeg').

To display your image, use the MATLAB command imshow( ). Since we will be working with gray scale images in this lab, your next step is to convert your input image to an 8-bit gray scale format using the rgb2gray( ) command. Display the resulting image. Check its size using the size( ) command. Use the title command to add a comment to your image. In your report, include the 8-bit grayscale image and specify its dimensions.

Next, you will perform edge detection on your image. (Edge detection is often a first step used in more complicated image processing operations.) Two-dimensional convolution, appropriate for images, is implemented in MATLAB via the function conv2( ). Convolution can be used to implement edge detection. Create the following Sobel vertical edge detection convolution kernel. This kernel is designed to respond maximally to edges running vertically relative to the pixel grid. It is a two-dimensional matrix $h_1[n,m]$, in MATLAB notation:

h1 = [-1 0 1; -2 0 2; -1 0 1]

Next create the following Sobel horizontal edge detection convolution kernel. This kernel is designed to respond maximally to edges running horizontally relative to the pixel grid. It is a two-dimensional matrix $h_2[n,m]$, in MATLAB notation:

h2 = [1 2 1; 0 0 0; -1 –2 -1]

Now, convolve your grayscale DailyShow image with the two edge detection kernels described as follows.

Assume M1 is the result of convolving the grayscale DailyShow image with h1 (i.e. M1 is the row gradient of the grayscale DailyShow image), and M2 is the result of convolving the grayscale DailyShow image with h2 (i.e. M2 is the column gradient of the grayscale DailyShow image). Use MATLAB to display the row gradient magnitude (|M1|), the column gradient magnitude (|M2|), and the overall gradient magnitude (i.e. (M1^2 + M2^2)^0.5).

Include these edge images (i.e. |M1|, |M2| and (M1^2+ M2^2)^0.5) in your report. Save your MATLAB function in an m-file. Include this file in your E-Submit.

[Optional]

You may notice lots of dark areas in these edge images. To save the toner (or cartridge) of your printer, you can try to figure out a way to reverse the grayscale of your edge images before printing them out. That is, you do a transformation to map the original darkest area to the brightest one, and the original brightest area (e.g. edge) to the darkest one.

3. Using Your Own Images

To better understand the horizontal and vertical masks, you will now use your own image. Find an image that you like, one of your photographs or some other .jpg file off the web, which either has a lot of horizontal or vertical edges in it (or both).

Assignment 2

Choose an edge detector to apply to your image (horizontal if your image has horizontal edges, or vertical if your image has a lot of vertical edges, or both) to your image. Save your original image and the edge images you create. Include the original image and the edge images in your report. You may display the direct result of the convolution, or the edge gradient magnitudes as in Assignment 1. Either way, explicitly state your choice in your write-up.

4. Scaling

In this section you will investigate scaling of images in the spatial domain. You will scale the image X[n,m] in both the vertical and horizontal directions using the same scaling factor S. An example where you want to use such scaling would be creating thumbnail-sized pictures for a web page.

Assignment 3

Write a MATLAB function that has an input argument for the scaling factor S. The function should read in the color DailyShow image, convert it to a gray scale, and then shrink the image by the scaling factor to form a thumbnail image. You are going to scale the original image with scaling factors S = 2 and S = 5. First, perform a simple scaling – keep one out of S^2 pixels. Since this is a 2D scaling, you can keep the center pixel in each square of S^2 pixels when S is an odd number, and one of the 4 center pixels when S is even.

Next, you are going to perform a more advanced scaling operation. Instead of keeping the center pixel in each square of S^2 pixels, keep the average of all of the pixels in this square. Display all four results in your report. Be sure to label each image with the scale factor used to create it. For S = 2 and S = 5 respectively, compare the two scaled versions of the original picture (i.e. picking one out of S^2 pixels versus picking the average of each block with S^2 pixels). Which one is the better thumbnail image?

Assignment 4

Guess how the following images look like when compared to the original image X[n,m]

(i) $X[N-n+1, m]$
(ii) $X[n, M-m+1]$
(iii) $X[N-n+1, M-m+1]$
where $1 \le n \le N,\ 1 \le m \le M$

Use DailyShow.jpg as the original image (i.e. X[n,m]). Verify your guesses by displaying resulting images of (i) (ii) (iii) in your report. Use help fliplr( ) and flipud( ) to see more information.

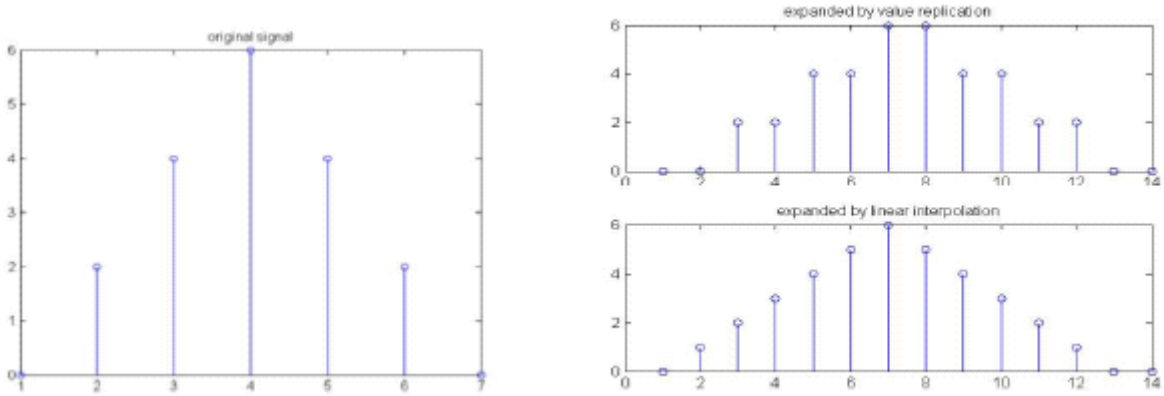
Assignment 5

When an image is scaled up to a larger size, there is the question of what to do with the new spaces in between the original pixels. Taking a 1-dimensional signal [0, 2 ,4, 6, 4, 2, 0] for example, suppose now we want to expand it by a factor of 2, then you can think of two common ways:

(i) simply double each sample – value replication; and

(ii) make the new samples half way between the previous samples – 2 tap linear interpolation.

The original signal and the up-scaled signals from the two methods are shown below.



Now, write a MATLAB function that can expand the input image (DailyShow.jpg) with dimension NxM to a 2Nx2M image using linear interpolation. Display this 2Nx2M image in your report. Save your MATLAB function(s) in m-files and include these files in your E-Submit.

Hint: You can directly use interp2( ) provided by MATLAB. Figure 2 shows the concept of "bilinear interpolation" that is used to deal with a 2-dimensional signal. Also, you can type help interp2 to see more information.

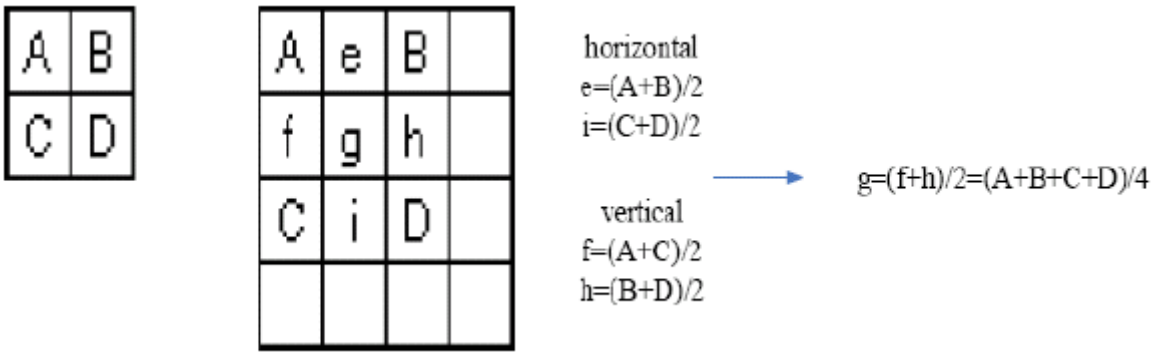


Figure 2. Enlarging the image by a factor of 2 using bilinear interpolation.