Computing for Scientists - Lab 4

CS 1340 — Dr. Mihail
Department of Mathematics and Computer Science
Valdosta State University

April 10, 2018

1 Introduction

In this lab you will learn to load images in MATLAB and do some basic image processing tasks. Images are made up of an organized collection (lattice) of pixels. Each pixel in an image has a coordinate (x, and y) as well as a color. The color is a 3-tuple of red, green and blue values (R, G, B). A color image is a 3D matrix of size h by w by 3, where h is the height of the image, w is the width, and 3 is the number of color channels (rgb).

You will use the MATLAB Image Processing Toolbox to load images in MATLAB. The function `imread(filename)` will load an image file into a matrix. For example: `im = imread('lena.png');` will load the image file called lena.png into a matrix `im`, assuming the file is in the same folder as your MATLAB script. Below is an example to read an image and display its size.

```
>> im = imread('lena.png');
>> size(im)

ans =

   512   512     3
```

Notice the third dimension of the matrix is 3. The first dimension contains the red channel, the second dimension contains the green channel and the third dimension contains the blue channel.

To display an image in a MATLAB figure, you can use the Image Processing Toolbox `image(image_matrix)` function.

For this lab, there will be one script for all problems. Make three different figures.

2 Problem 1 (20 points)

You will load the image and display each color channel separately. To do this, you will first load the image into a matrix, and make three copies of that matrix, called: `R`, `G`, and `B`. To

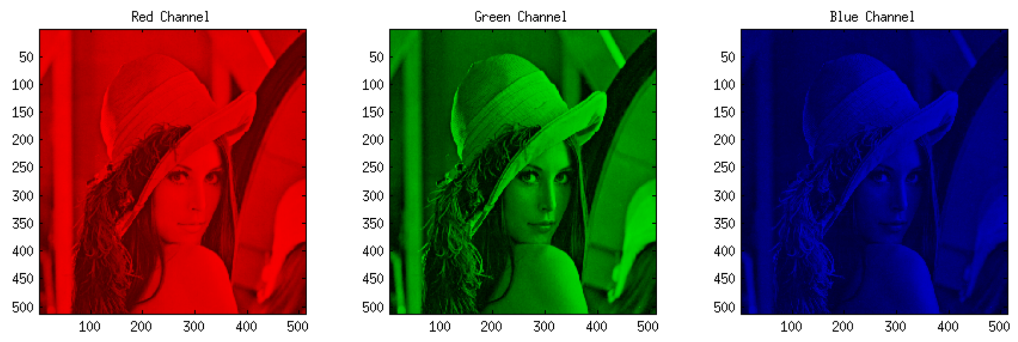

Figure 1: Red, green and blue channels displayed separately

display individual channels you must set the other channels to 0. For example, the second and third dimensions of the `R` matrix have to be set to 0 to display the red channel. The first and third dimensions of the `G` matrix have to be set to 0 to display the green channel. The first and second dimensions of the `B` matrix have to be set to 0 to display the blue channel. The final result should look like Figure 1.

Write the code to make the Figure exactly as displayed above.

3 Problem 2 (20 points)

Next, you will extract individual channels using matrix slicing. The result should be a 2D matrix. For example, to extract the red channel you would do the following:

```
Rchannel = im(:, :, 1);
```

Do this for all three channels. You will then set the green channel to zero and combine the 2D matrices into one using MATLAB `cat` command:

```
nogreen = cat(3, Rchannel, Gchannel, Bchannel);
```

You have to set each matrix element to 0, not convert it to a scalar. In other words `Rchannel = 0` will not work. The above code will **concatenate** (combine) `Rchannel`, `Gchannel` and `Bchannel` into a matrix called `nogreen`, along the third dimension (the first argument of the function call). Your result should look like Figure 2:

4 Problem 3 (40 points)

For this problem, you will threshold the input image channels into 2 values: 0 or 255. You will look over each pixel, and code the following logic:

- If the red channel value is below 127, you will set it to 0, otherwise set it to 255.
- If the green channel value is below 127, you will set it to 0, otherwise set it to 255.
- If the blue channel value is below 127, you will set it to 0, otherwise set it to 255.

The result should look like Figure 3.

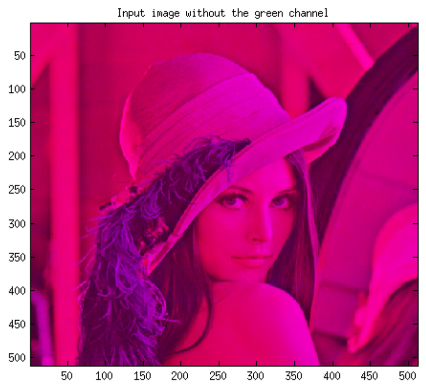

Figure 2: Image without the green channel.

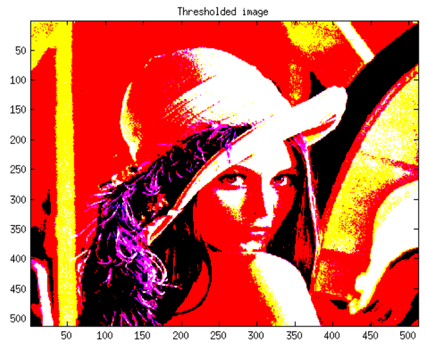

Figure 3: Thresholded color image.

Due dates

MATLAB script files due before midnight on Sunday, April 15th.