**ELECTIVE 3**

Midterm Exam

**Image Processing in Octave**

|  |
| --- |
|  |

Score

*Submitted by:*

**De Vera, Giovanni**

**Thursday 3:00pm – 4:30pm / 4-2**

*Date Submitted*

**24-11-2022**

*Submitted to:*

**Engr. Maria Rizette H. Sayo**

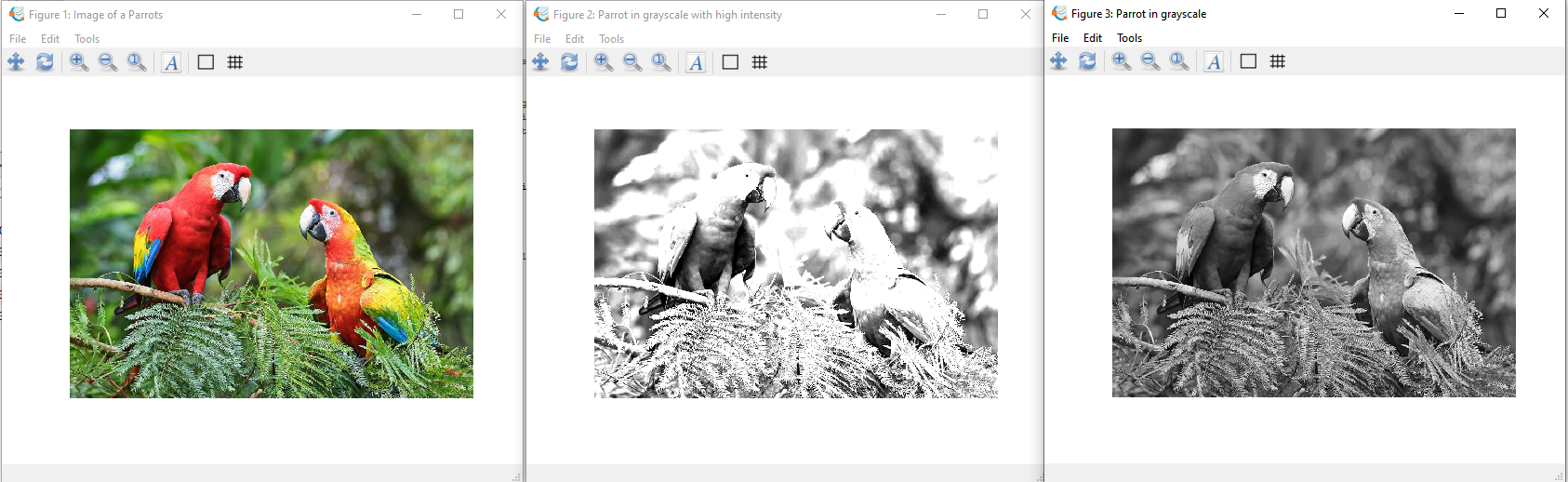
Methodology

1. Importing, Displaying, and Converting Images
2. Using the Editor of Octave, create a program that will load and display parrots.jpg
3. Examine the size of the parrots by typing whos to find out the size of the image that you have read in
4. Convert the class uint8 color image parrots to a gray scale image, and display the full intensity range gray-scale image using the imshow command
5. Covert the true color image to a gray-scale image
6. Save the program to this format parrots.m
7. Display of Color Images
8. Open the image file nature.jpg from the source folder
9. Read in the file nature.jpg and display it on the screen as a reference image How large an image is created when we use the RGB representation compared to a gray-scale image conversion of it?
10. Assign an image color that intensifies red, green, and blue and display each image in one window
11. Convert each image file extension to png
12. Save the program as nature.jpg

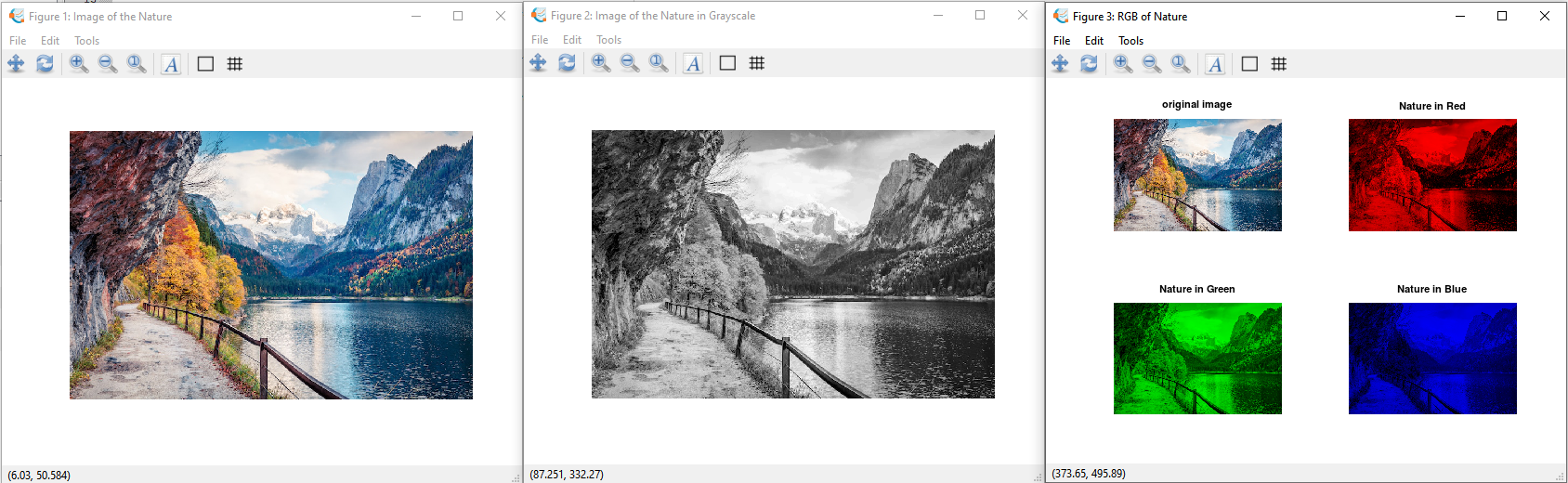
*Note: Take a screen shot of the output display of the converted images and do not forget to write your Octave code to this manuscript*

Screen shot of Output Image

1.



2.



Source Codes:



# To clear all variables and figures

clc

clf

clear all

close all

# a.

parrots = imread("parrots.jpg");

figure(1, "name", "Image of a Parrots"): imshow(parrots);

# b - 533x800x3

whos parrots # To show the size of the original image of nature

# c

pkg load image; # importing a package

grayscale\_parrot = rgb2gray(parrots); # changing the image to grayscale

high\_intesity\_parrot = imadjust(grayscale\_parrot); # adjusting the image

high\_intesity\_parrot = rescale(high\_intesity\_parrot, 0, 255); # rescaling the image

high\_intesity\_parrot = 2 \* mat2gray(high\_intesity\_parrot); # value of the intesity

# Display and naming the figure window

figure(2, "name", "Parrot in grayscale with high intensity"): imshow(high\_intesity\_parrot);

# d

parrot\_grayscale = rgb2gray(parrots); # changing the image to grayscale

# Display and naming the figure window

figure(3, "name", "Parrot in grayscale"): imshow(parrot\_grayscale);

2.

# To clear all variables and figures

clc

clf

clear all

close all

# b.

nature = imread("nature.jpg");

# Display and naming the figure window

figure(1, "name", "Image of the Nature"): imshow(nature);

whos nature; # To show the size of the original image of nature

nature\_grayscale = rgb2gray(nature); # Convert true color to grayscale

# Display and naming the figure window

figure(2, "name", "Image of the Nature in Grayscale"): imshow(nature\_grayscale);

whos nature\_grayscale; # To show the size of the nature in grayscale.

# c.

# To intensify the red color

nature\_red = nature;

# Set Blue and Green to 0

nature\_red(:,:,2)=0;

nature\_red(:,:,3)=0;

# To intensify the green color

nature\_green = nature;

# Set Red and Blue to 0

nature\_green(:,:,1)=0;

nature\_green(:,:,3)=0;

# To intensify the blue color

nature\_blue = nature;

# Set Red and Green to 0

nature\_blue(:,:,1)=0;

nature\_blue(:,:,2)=0;

# Display and naming the figure window

figure(3, "name", "RGB of Nature"),

# Plotting the original image and Red, Blue, Green image of nature

subplot(2,2,1),imshow(nature),title('original image');

subplot(2,2,2),imshow(nature\_red),title('Nature in Red');

subplot(2,2,3),imshow(nature\_green),title('Nature in Green');

subplot(2,2,4),imshow(nature\_blue),title('Nature in Blue');

# d.

# Converting all the images to PNG file.

imwrite(nature, "nature\_in\_grayscale\_in\_PNG.png");

imwrite(nature\_grayscale, "nature\_in\_grayscale\_in\_PNG.png");

imwrite(nature\_red, "nature\_red.png");

imwrite(nature\_green, "nature\_green.png");

imwrite(nature\_blue, "nature\_blue.png");

Conclusion

*My conclusion for this exam is it helps me to understand how an image can be manipulated. It also helps to know the basic commands and behaviour of every function when it comes to manipulating an image.*

Rubrics in Grading the Midterm Exam

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A – Excellent | B – Good | C – Fair | D – Needs Improvement |
| Specifications | The Program works and meets all of the specifications | The program works and produces the correct results and displays them correctly. It also meets the most of the other specifications | The program produces correct results but does not display them correctly | The program is producing incorrect results |
| Readability | The code is exceptionally well organized and very easy to follow | The code is fairly easy to read | The code is readable only by someone who knows what it is supposed to be doing | The code is poorly organized and very difficult to read. |
| Reusability | The code could be reused as a whole or each routine could be reused | Most of the code could be reused in other programs | Some parts of the code could be reused in other programs | The code is not organized for reusability |
| Documentation | The documentation is well written and clearly explains what the code is accomplishing and how | The documentation consists of embedded comment and some simple header documentation that is somewhat useful | The documentation is simply comments embedded in the code with some simple header comments separating routines | The documentation is simply comments embedded in the code and does not help the reader understand the codes |
| Efficiency | The code is extremely efficient without sacrificing readability and understanding. | The code is fairly efficient without sacrificing readability and understanding | The code is brute force and unnecessarily long | The code is huge and appears to be patched together |
| TOTAL |  | | | |

https://github.com/DeVeraGiovanni/CPEN111