.

Title	Image Effect Implementation, Part:1
Student Name	Jubayer Hossain Arnob
Student ID	1813124642
Email address	jubayer.arnob@northsouth.edu

Question

Implement and explain your approach to implement the following image effects. **(a)** Posterize **(b)** Nightvision **(c)** Photocopy **(d)** Vignetting **(e)** Mirror Image.

Instructions

- Both explanations and Matlab codes should be included in **a single** document. No need to submit separate files for codes.
- Include some input and output of your code. In addition to the sample input/output (from code\SampleIN-OUT), also include more input/output using other photos of your choice.
- You can also include sample output of the intermediate steps of your calculation to explain the reason/effect with/without any particular step.
- Note that, only producing the output is not the only end goal here. I will also mark your understanding of the work from your explanation. Try to illustrate as elegant as you can with figures, tables, algorithms, charts, etc. like a research paper.
- Plagiarism of any type are strictly prohibited.

Solution Hint (These hints are discussed in detail during online class)

(a) Posterize: Usually values within [0,255] represent a pixel. Instead of using 256 colors, you have to use less number of colors like 4-8. Every pixel will get the closest output color.
(b) Nightvision: Output R = Input G / 2, Output B = 2 x Output R, Output G = 2 x Output B
(c) Photocopy: Convert the input to gray level image. Then, consider a threshold = 100. If a pixel value is greater than the threshold, put 255 as the color value in the output. Otherwise, put input pixel x (threshold - input pixel) / (threshold^2) as the output pixel.
(d) Vignetting: Calculate the center pixel of the image and the maximum distance (M) possible from the center to any corner. Then, go to each pixel and calculate the distance (D) from that pixel location to the center. Find a darkness weight = 1 - D / M. For the output pixel, multiply the weights with the input pixel value. Note: try to find a way to vary the darkness level based on controlling a parameter.
(e) Mirror Image: We can get the mirror image of the given image if we reverse the order of the pixels (elements of the matrix) in each row.

Introduction

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. In this report I am going to discuss about assignment 1 that includes 5 image processing tasks. Challenges are to give the following effects (Posterize, Night vision, Photocopy, Vignette and Mirror) on an input image using MATLAB tool. My approach on the implementation is explained here.

Method

This tasks in this assignment are done in MATLAB ver. R2021b. By understanding the problems, I have to get the idea how to solve the problem and go for the code. So, one thing is similar for all the tasks that I have to read an input image and show the output image along with original. Now the approaches to solve the tasks will be discussed here.

- First task is to posterize the input image. As we know, pixel value is represented from 0 to 255. That eventually are 256 colors. So, if we decrease the number of pixel value from 256 to 4-8 it will show less color and should make the image look like poster image. So, to make this happen, I have chosen particular nearby values and made it one value. For example, values from 0 to 50 will be represented by 20. So, 50 colors are now only one color. This is how posterization effect will be given till 255 no. value. The color gaps will be around equal distance. Figure 1 shows the expected outcome.

Figure:1

- Next task here is to give night vision effect on the input image. Here I have followed a formula where the green channel of input will be divided by 2 and goes into red channel of output. After that the red channel of input gets multiplied by 2 and goes into blue channel of output. The blue channel of output becomes input that is multiplied by 2 and goes into green channel of output. Formula: Output R = Input G / 2, Output B = 2 x Output R, Output G = 2 x Output B. Figure 2 shows the expected outcome following the method.

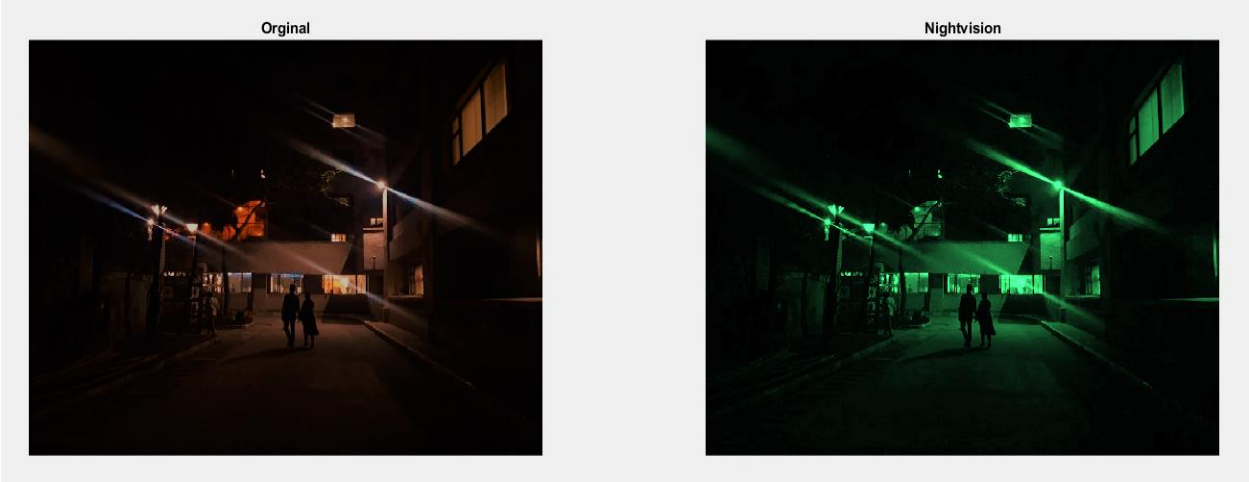

Figure:2

- Third task is photocopy effect. For this I have to convert it to gray level first. As it's a photocopy effect it has to be B&W And a threshold value has to be defined. What I have to do is, to see if the pixels are above the threshold value or lower than that. If higher I make it white which means change the value to 255. And if not, I will follow this formula to change the pixel value. Formula: output pixel = input pixel x (threshold - input pixel) / (threshold^2). Figure 3 is the outcome we expect by this method.

Figure:3

- For vignetting effect, first I have to calculate the center value as we know vignetting effect is circular from center to distance, more the distance more the dark effect. So, from this we understand distance is important factor here. After calculating center pixel value, I have to get the distance from one corner to the center. Distance here I measured from (1,1) point to center point. This distance is max distance from center. After that, I have calculated distance from center to a each and every point in image. Calculate the weight for each which is then multiplied with the particular pixel value. Weight calculation formula is, weight = 1- (that specific points distance from center / max distance from center). Figure 4 is the expected outcome for this method.

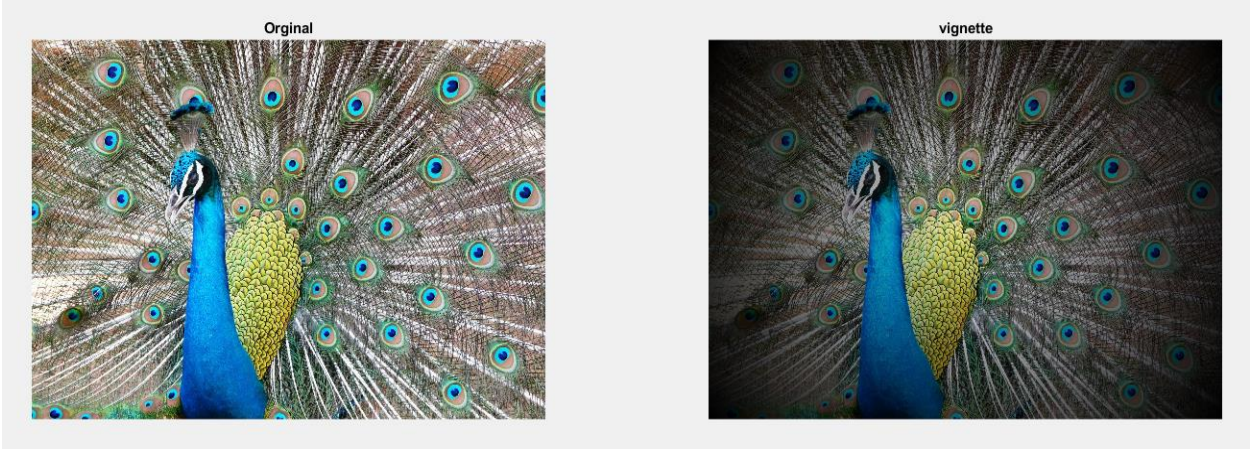

Figure:4

- The last task that is mirroring effect is simple to understand. If you want to mirror vertically you have to swap row values from far to close. To simplify suppose, a row has dimension of 100. So 100th value will swap with 1st value. 2nd value will swap with 100-1th value, 3rd value will swap with 100-2th value like this 50th value will swap with 100t-49th value. This will make it mirror. Same goes for horizontal mirror effect that is column position value swap. Figure 5 is what I expect by the method.

Figure:5

Experiment

Experiment 1(Posterize)

1st approach to see if the method works. So, I did a simple code which is in figure 6. And it worked.

Now the motivation is to give the power to the user to give the number of colors as input to get the output result. To make it look like a posterize image user must choose the value from 4 to 8 any value above it doesn't feel much of a posterize image also less than 4 colors destroy the posterize effect. If it is zero it will show 99% of the pixels as white and obviously giving input 255 will be the same image as original one. Figure 7 and 8 shows the posterize effect with input number of colors 4 and 8. Following.

```
f = im(i,j,k);
if f<=70
    im(i,j,k) = 50;
elseif f>70 && f<=120
    im(i,j,k) = 100;
elseif f>120 && f<=180
    im(i,j,k) = 150;
else
    im(i,j,k) = 200;
end
```

figure: 6

Figure 7: Posterize with number of colors input: 4

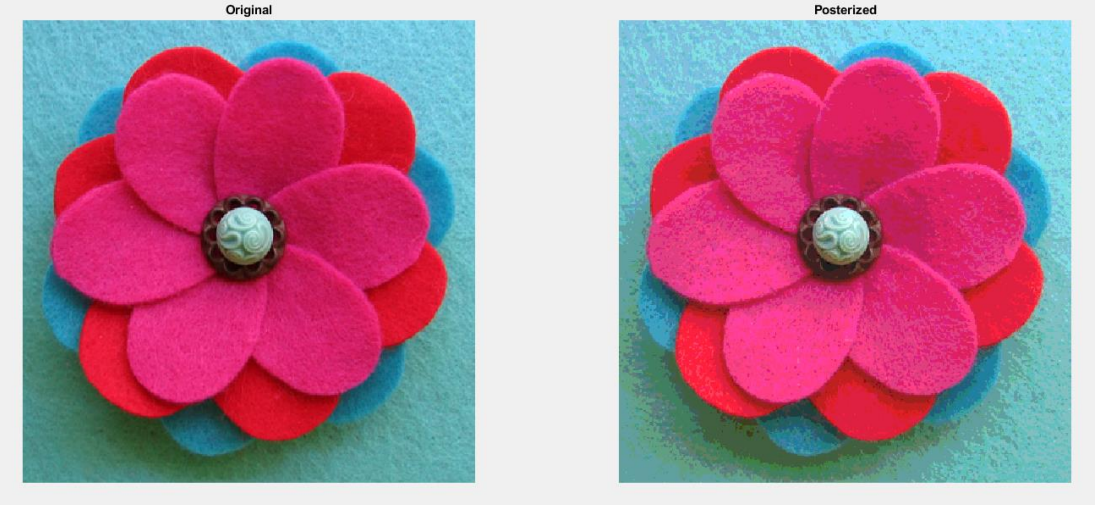

Figure 8: Posterize with number of colors input: 8

*Experiment 2(Night Vision)**:***

The formula in the method is implemented in code. This will put greenish appearance on the image that is known night vision effect. 2 is the best value that is found by researchers. Tried putting value less than 2 and more than 2 it makes the image darker and brighter following, also causes more fade or noise. So, figure 9 and 10 shows the result of the program.

Figure 9: Night-vision effect sample 1

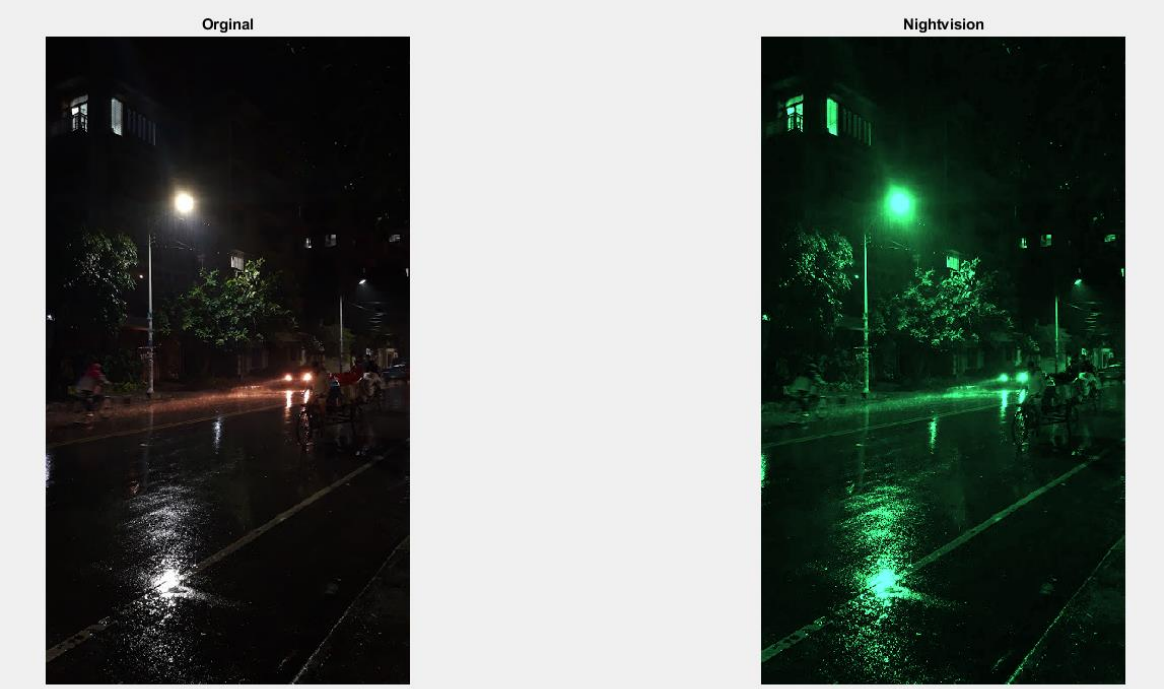

Figure 10: Night-vision effect on night city in rain

Experiment 3 (Photocopy):

The photocopy effect is implemented as following the description in methods. To convert RGB image to gray level image, here I have used the function “RGB2GRAY”. The threshold value here determines the outcome which is adjustable too for the users so that one can choose one’s

preferable outcome. Figure 11 and 12 shows the output of same image with 3 different threshold values 100, 80 and 120 following. As you can see, when your value goes down it becomes more whitish less detailed. And when the values go up it becomes more detailed and darker but at certain value later it will lose its details and appear as black ink image.

Figure 11: photocopy effect with threshold value 100

Figure 12: photocopy effect with threshold value 80 & 120 following

Experiment 4 (vignetting effect):

Vignetting effect is implemented following the method I have discussed earlier. To implement, I first found center row and column value. That shows the center point and from (1,1) point which is the left top corner point I have calculated the max distance from center and as calculated all the points in iteration and found the weight and multiplied it with corresponding pixel thus creates vignetting effect to the image. Figure 13 shows the output.

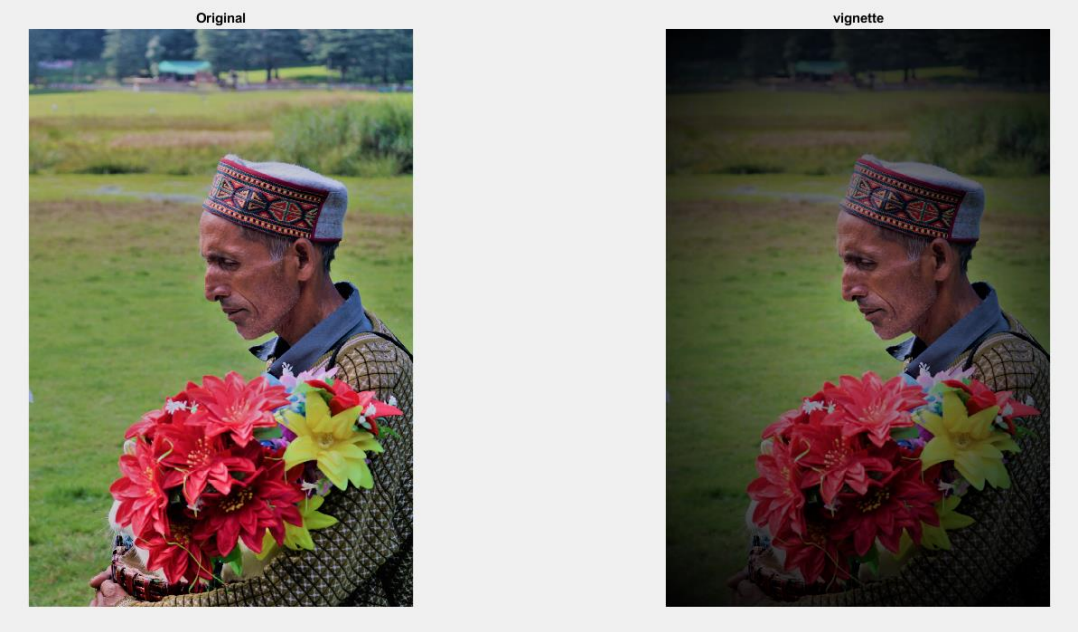

Figure 13: vignetting effect

By changing the k values in code, we can try blue, red or green color mixture in vignetting effect. For example, here in figure 14 I have put code K = 3:3 to make it greener which supposed to be dark followingly the general vignette effect. Or If I have use k = 2:3 it would be red around the center.

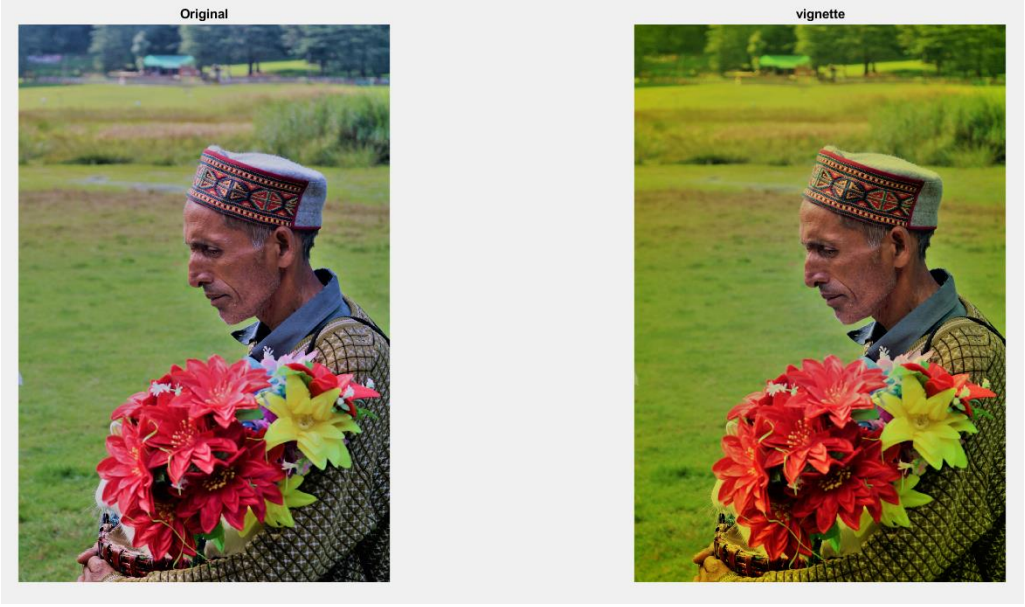

Figure 14: vignetting effect with rgb channel manipulation

Or another scope of experimenting here is to change the spot light. For example, here in figure 15 the vignette effect spot light isn't in a proper place. So, we can placement the center spot where we want to focus and make the image output better. In this figure I have to manipulate m, n value which are center row and center column value. By seeing the picture, I assumed center column is okay and I have to change the center row value from half to one third. Thus, the center value almost positioned where I wanted to spot. Figure 16 shows it.

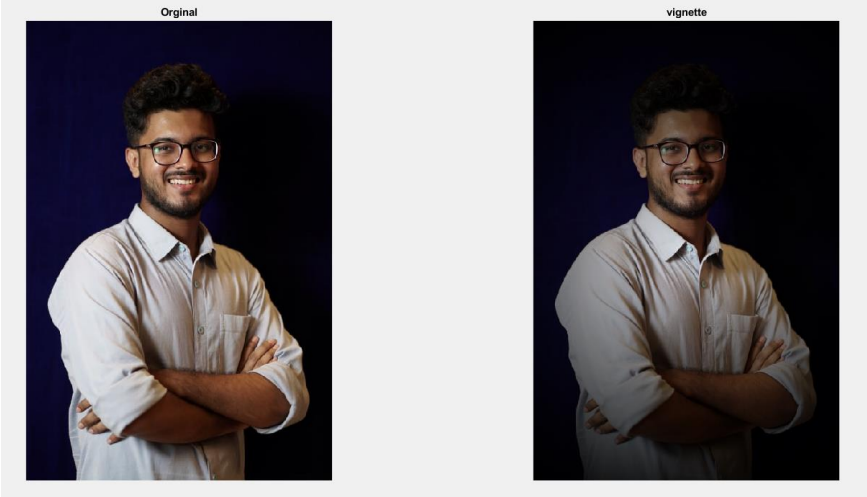

Figure 15: Vignette effect center focused.

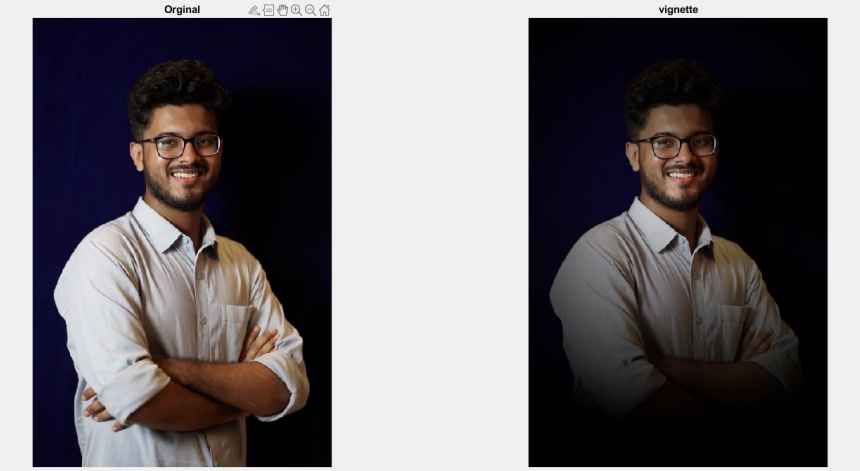

Figure 16: vignette effect manually put the spot to focus as center

Experiment 5 (Mirror):

The mirror effect is simple to implement. I had to swap row position or column position according to mirror type. Here I iterated values row, column and channel wise and changed the 1st Colum or row position with dimension size position. And per iteration positions were swapping with it's corresponding swap value that is size-1 per each iteration. So here I let the user to select which mirror type he or she prefers. If 1 vertical mirror and for 2 horizontal mirror as code. Figures shows mirror effect.

Figure: vertical mirror effect

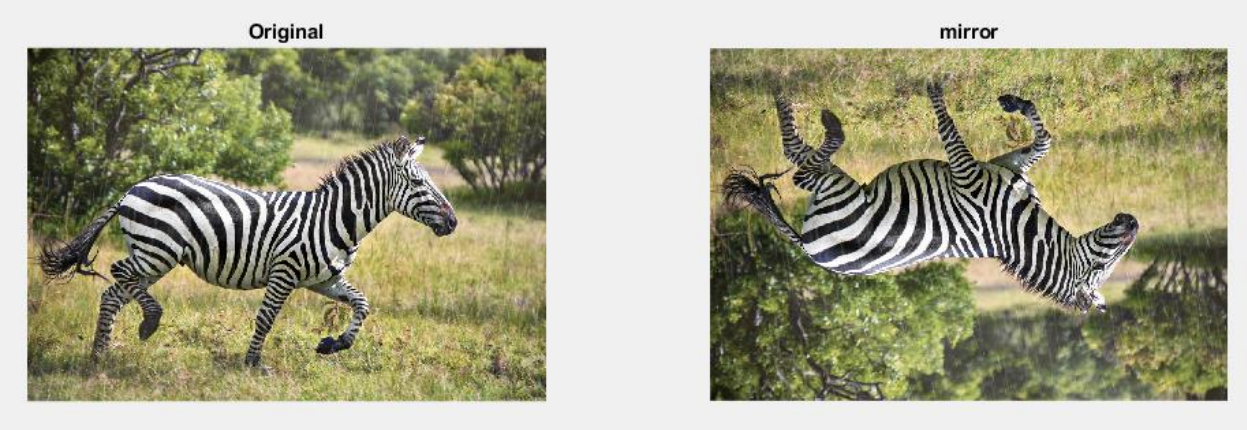

Figure: horizontal mirror effect

Conclusion

The tasks in this assignment have been really interesting from solving problems to trying new things to experiment and manipulate images. All though there are some issues where one must be careful while doing the tasks. A very important thing is the dimension start here from (1,1) not (0,0). This may create issue in calculating and output value. I faced error so I understood the problem and solved it. Using MATLAB tool is a new experience and a good environment to play around and experimenting. I believe after doing this assignment I have enough motivation and interest to learn more and do projects and paper writing in image processing sector.

Reference

- "Solution Hint" Lecture video 2
- "A Beginner's Guide to MATLAB" Beginners_guide_to_MATLAB.pdf - Google Drive

Appendix

Code: Posterize

```
clc
clear all
close all

original = imread('horse_in.jpg');
im = original;
%imshow(im);
input = input('Number of colors from 4 to 8: ');
if input<4 || input >8
    disp("To make it posterize, You need to select number of colors from 4 to 8. "
+ ...
        "but you can try different values to experiment it");
end
for i = 1 : size(im,1)
    for j = 1 : size(im,2)
        for k = 1 : 3
            f = im(i,j,k);
            for a = 1:input
               m = round(255/input);
               n=m*(a-1);
               m=m*a;
               if f>n && f<=m
                    im(i,j,k)=m;
               end
            end
        end
    end
end
subplot(1,2,1)
imshow(original); title('Original')
subplot(1,2,2)
imshow(im); title('Posterized')
```

Code: Nightvision

```
clc
clear all
close all
original = imread('rain_in.jpg');
im = orginal;
%imshow(im);
im(:,:,1) = im(:,:,2)/2;
im(:,:,3) = 2*im(:,:,1);
im(:,:,2) = 2*im(:,:,3);

subplot(1,2,1)
imshow(original); title('Original')
subplot(1,2,2)
imshow(im); title('Nightvision')
```

Code: Photocopy

```
clc
clear all
close all

original = imread('horse_in.jpg');
im = rgb2gray(original);
%imshow(im);

th = input("Threshold value: ");
for i = 1 : size(im,1)
    for j = 1 : size(im,2)
        if im(i,j)>th
            im(i,j)= 250;
        else
            im(i,j) = im(i,j)*(th-im(i,j))/(th*th);
        end
    end
end

subplot(1,2,1)
imshow(original); title('Original')
subplot(1,2,2)
imshow(im); title('Photocopy')
```

Code: Vignette

```
clc
clear all
close all

original = imread('flowerman.jpg');
im = original;
imshow(im);
m = (size(im,1)/2);
n = (size(im,2)/2);
maxDist= (sqrt((m-1)^2+(n-1)^2));

for i = 1 : size(im,1)
    for j = 1 : size(im,2)
        for k = 1:3
            ranDist = (sqrt((m-i)^2+(n-j)^2));
            weight= 1-(ranDist/maxDist);
            im(i,j,k)=im(i,j,k)*weight;
        end
    end
end

subplot(1,2,1)
imshow(original); title('Original')
subplot(1,2,2)
imshow(im); title('vignette')
```

Code: Mirror

```
clc
clear all
close all

im = imread('mirror_in.jpg');
%imshow(im);

mirror_type = input("If horizontal mirror put 1, if verticle mirror put 2: ");
v = (size(im,2))+1;
h = (size(im,1))+1;

for i = 1 : size(im,1)
    for j = 1 : size(im,2)
        for k = 1:3
            if mirror_type == 1
                im2(i,j,k) = im(i,v-j,k);
            elseif mirror_type == 2
                im2(i,j,k) = im(h-i,j,k);
            end
        end
    end
end

subplot(1,2,1)
imshow(im); title('Original')
subplot(1,2,2)
imshow(im2); title('mirror')
```