Experiment No.: 01

Experiment Name: Experimental Study of Familiarization with Fundamental MATLAB Commands Used in DIP.

Objectives:

- To know about different common image format.
- To know about different types of images.
- To call, read, write and show an image using MATLAB commands.
- To convert an RGB image into gray.
- To find the complementary of an image.

1.1 Theory:

Digital image processing is the use of computer algorithms to create, process, communicate, and display digital images. It plays an important role both in daily life applications as well as in the areas of research technology. MATLAB provides a comprehensive set of reference-standard algorithms and workflow apps for image processing, analysis, visualization, and algorithm development. The different elements of an image processing system include image acquisition, image storage, image processing and display.

Table-01: Basic MATLAB commands

SL. No.	Name	Functions
a)	imread( )	Read image from graphics file.
b)	imwrite ( )	Write image to graphics file.
c)	imshow( )	Display image in a figure.
d)	imfinfo( )	Information about graphics file.
e)	imagesc( )	Display image with scaled colors.
f)	imcomplement( )	Complement image.
g)	colormap( )	View and set current colormap.

Table-02: Common image format

SL. No.	Name
a)	Graphics Interchange Format (GIF)
b)	Joint Photographic Experts Group (JPEG)
c)	Bit Map Picture (BMP)
d)	Portable Network Graphics(PNG)
e)	Tagged Image Format(TIF/TIFF)

Table- 03: Types of images

SL. No.	Name	Description
a)	RGB	Combination of red, green ,blue
b)	Grayscale	Intermediate state between RGB and Binary
c)	Binary	Black(0)/White(1)

Table- 04: Some built-in images in MATLAB

SL. No.	Name
a)	cameraman.tif
b)	onion.png
c)	wombats.tif
d)	autumn.tif
e)	spine.tif
f)	cell.tif

1.2 Equipment:

- Computer
- MATLAB Software
- Images

1.3 Problems:

1.3.1 Call, Read, Write and Show cameraman.tif.

Code:

```
clc;
clear;
img = imread('cameraman.tif');
imwrite(img,'example.png');
imshow('example.png')
```

Output:

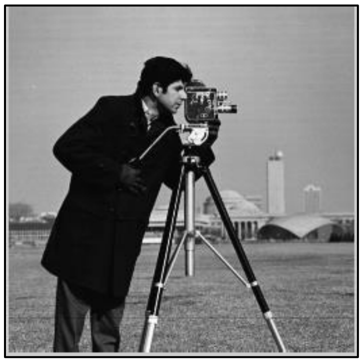

Fig 01: MATLAB built-in image (cameraman.tif)

1.3.2 Colormap(gray), Plot imwrite Array.

Code:

```
clc;
clear;

img= imread('onion.png');
imwrite(img,'example.png');
imshow('example.png')

colormap(gray);
```

Output:

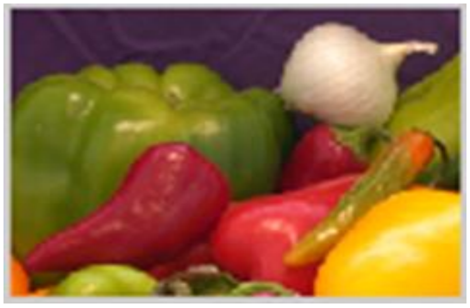

Fig 02: MATLAB built-in image (onion.png)

1.3.3 Read image cell.tif(gray), spine.tif(jet), onion.tif(gray) and subplot them in a single picture.

Code:

```
clc;
clear;

img1 = imread('cell.tif')
img2 = imread('spine.tif')
img3 = imread('onion.png')

ax(1) = subplot(1,3,1), imshow(img1);
ax(2) = subplot(1,3,2), imshow(img2);
ax(3) = subplot(1,3,3), imshow(img3);

colormap(ax(1),gray);
colormap(ax(2),jet);
colormap(ax(3),gray);
```

Output:

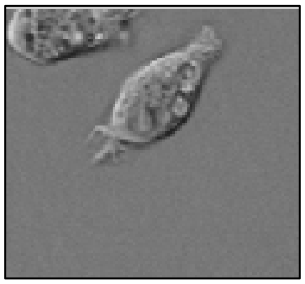

Fig 03: MATLAB built-in image (cell.tif)

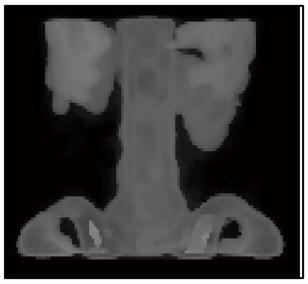

Fig 04: MATLAB built-in image (spine.tif)

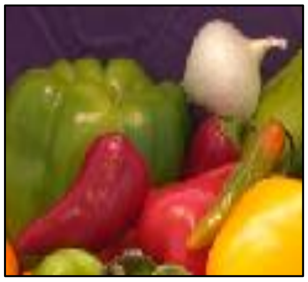

Fig 05: MATLAB built-in image (onion.png)

1.3.4 Read onion.png and convert it from RGB to gray.

Code:

```
clc;
clear;

taken_image = imread('onion.png');
gray = rgb2gray(taken_image);

imshow(gray)
```

Output:

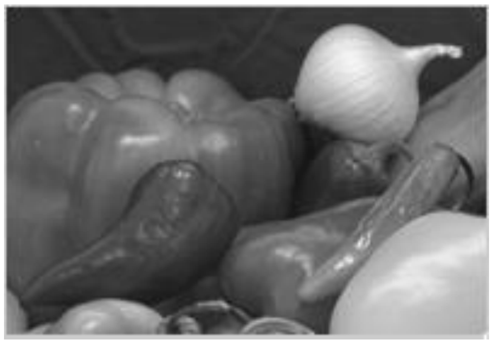

Fig 06: Image RGB to Gray Conversion (onion.png)

1.3.5 In a frame, subplot original, red, green, blue image.

Code:

```
clc;
clear;

rgbImage = imread('onion.png');

% Extract color channels.
redChannel = rgbImage(:,:,1);
greenChannel = rgbImage(:,:,2);
blueChannel = rgbImage(:,:,3);

% Create an all black channel.
allBlack = zeros(size(rgbImage, 1),size(rgbImage, 2), 'uint8');
just_red = cat(3, redChannel, allBlack, allBlack);
just_green = cat(3, allBlack, greenChannel, allBlack);
just_blue = cat(3, allBlack, allBlack, blueChannel);

%plot  all images
subplot(2, 1, 1);
imshow(rgbImage);
fontSize = 20;
title('ORIGINAL IMAGE', 'FontSize', fontSize);

subplot(2, 3, 4);
imshow(just_red);
title('RED', 'FontSize', fontSize);

subplot(2, 3, 5);
imshow(just_green);
title('GREEN', 'FontSize', fontSize);

subplot(2, 3, 6);
imshow(just_blue);
title('BLUE', 'FontSize', fontSize);
```

Output:

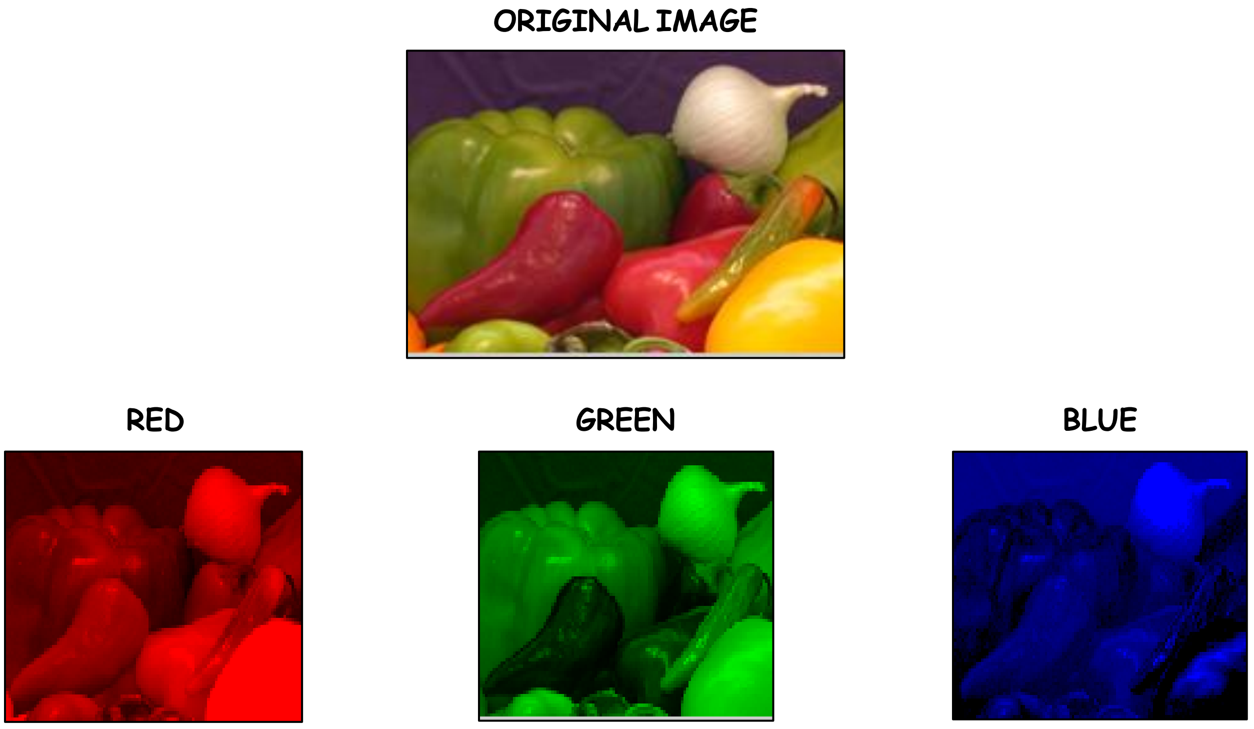

Fig 07: Image Conversion RED, GREEN, BLUE (onion.png)

1.3.6 Find the Complementary of an Image.

Code:

```
I = imread('cameraman.tif');
J = imcomplement(I);
imshowpair(I, J, 'montage');
```

Output:

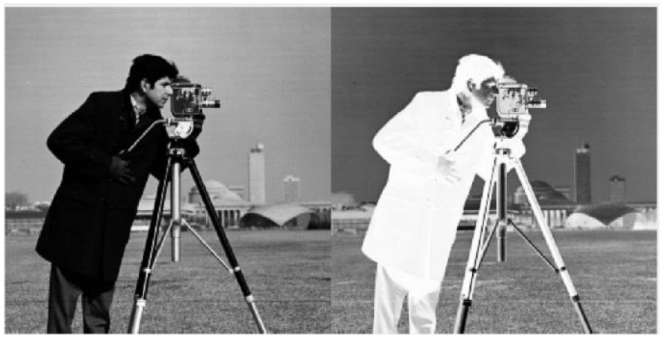

Fig 08: Complementary of an Image

1.4 Discussion:

In this experiment we have learnt the very basic of Digital Image Processing using MATLAB software. Firstly, knowledge about some common inbuilt image and function was given. These functions were then used for various basic image processing. It is to mention that colormap function didn’t worked on ‘onion.png’ image. Later, ‘onion.png’ image was converted from RGB to gray color using MATLAB function. Another problem occur during experiment was the MATLAB software could not call pictures form local storage of the computer. So, here for Experiment 6 I have used a prebuilt image (cameraman.tif). It is also possible to change an original image into red, green, blue color using MATLAB. Finally, complement of a given image was found and shown in the MATLAB software using complement function.