CLASS	**: B.E. E &TC**	**SUBJECT: DIVP**
EXPT. NO.	**: 10**	**DATE:16-10-2020**

TITLE : ENHANCE A GRAY IMAGE USING PSEUDO COLORING

CO 1:	Apply the fundamentals of digital image processing to perform various operations on an image-enhancement in spatial domain/ frequency domain, image-restoration, image compression, video filtering and video compression on a given gray image. Examine the effect of varying the mask size and density of noise in an image and comment on the obtained results.
CO4:	Carry out experiments as an individual and in a team, comprehend and write a laboratory record and draw conclusions at a technical level.

AIM :

To implement:

- **To load gray image**
- **To perform histogram equalization on the gray image**
- **Assigning a specific color to a range of pixels have same gray intensity level**

SOFTWARES REQUIRED: Matlab 7.0 or above, python

THEORY:

10.1 Pseudo coloring

Pseudo-color processing is a technique that maps each of the grey levels of a black and white image into an assigned color. This colored image, when displayed, can make the identification of certain features easier for the observer. The mappings are computationally simple and fast. This makes pseudo-color an attractive technique for use on digital image processing systems that are designed to be used in the interactive mode.

10.2 Intensity slicing

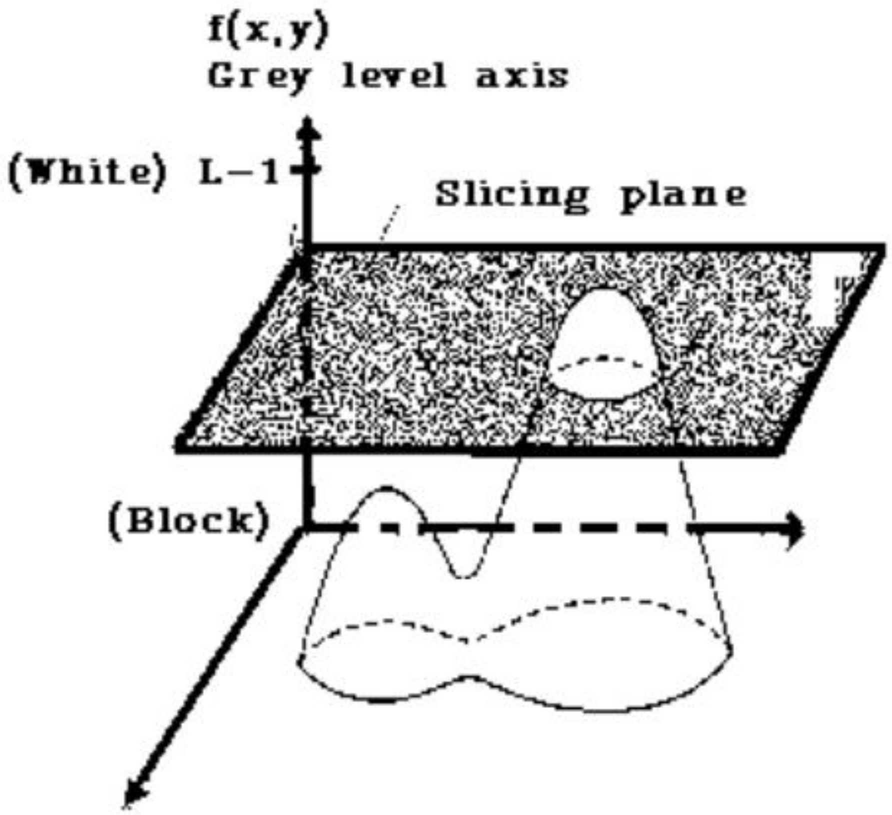


Fig. 1: Geometric interpretation of the intensity-slicing technique

The technique of density also called intensity slicing and color coding is the simplest example of pseudo color image processing. If an image is interpreted as a 3D function[1][2], the method can be viewed as one of placing planes parallel to the coordinate plane of the image. Each plane then slices the function in the area of intersection. The above figure shows a plane at f(x, y) = L_i to slice the image function into two levels. I a different color is assigned to each side of the plane shown in Fig.1, any pixel whose gray scale is above the plane will be coded with one color, and any pixel below the plane will be coded with the other. The result is a two-color image whose relative appearance can be controlled by moving the slicing plane up and down the gray level axis. The idea of planes is useful primarily for a geometric interpretation of the intensity-slicing technique. When more levels are used, the mapping function takes on a staircase form.

10.3 Gray Level to Color Transformations

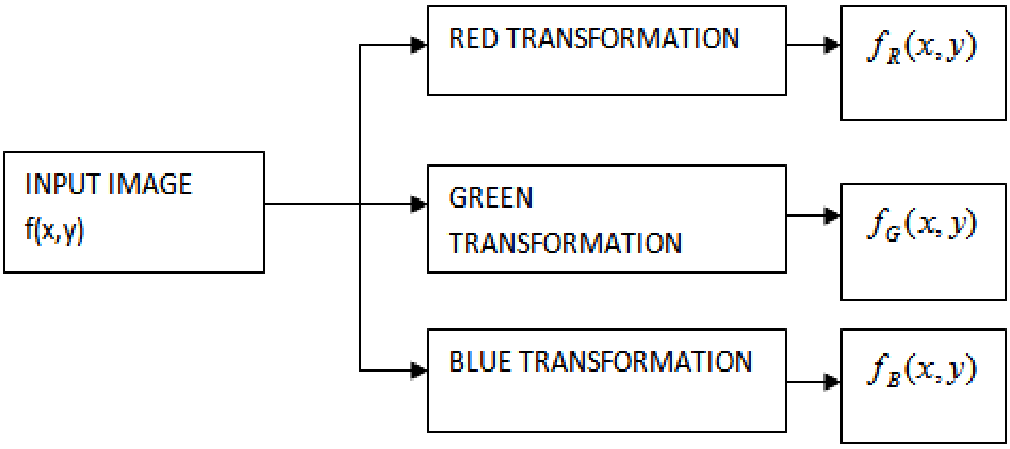


Fig 2: Functional block diagram for pseudo color image processing f_R, f_G, f_B are fed into the corresponding Red, Green, and Blue inputs of an RGB color monitor

There are many types of transformations and are capable of achieving a wider range of pseudo color enhancement results than the simple slicing technique discussed in the preceding section. An approach that is particularly attractive is shown in Fig.2. Basically the idea underlying this approach is to perform three independent transformations on the gray level of any input pixel. The three results are then fed separately into the red, green, and blue channels of a color television monitor. This method produces a composite image, whose color content is modulated by the nature of the transformations on the gray-level values of an image, and is not functions of position.

10.4 Need of pseudo coloring

Pseudo coloring results in better identification of these calcifications. The term Pseudo-color refers to using color to visualize things that aren't inherently colored. If two variables of information have to be displayed at a time in an image, the Proceedings of the International Conference on Cognition and Recognition 755 intensity (one variable) and the hue (another variable) may be varied independently. Pseudo -color can be effectively used to label or identify particularly significant sections of the image as determined by some other image processing or vision algorithm. It has to be kept in mind while using a color that a large portion of the population (estimated 10% of all males) is color blind. If isoluminant displays are created (one with uniform intensity and varying hues), color blind people can't see the displays. Normal sighted people will see them poorly.

10.5 Conclusion:

1. Pseudo coloring is a technique to artificially assign colors to a grey scale.
2. Pseudo Color images can help to reveal image qualities that would not be readily visible within the image's true color. The false color of a pixel is created by determined by summing its RGB values and mapping them into a 768-row lookup table.
3. It is one of the efficient color map manipulation technique.
4. The no of available colors gets limited in pseudo coloring or when private color map is used, colors may flash between different applications.

1.5 References:

i. Gonzalez R, Woods R, "Digital image processing", Pearson Prentice Hall, 2008.

ii. Gonzalez R, Woods R, Steven E, "Digital Image Processing Using MATLAB®", McGraw Hill Education,2010.

iii. https://pdfs.semanticscholar.org/edd3/72b3bce9968b8d8f93d2ffb52df3f5b96343.pdf

____

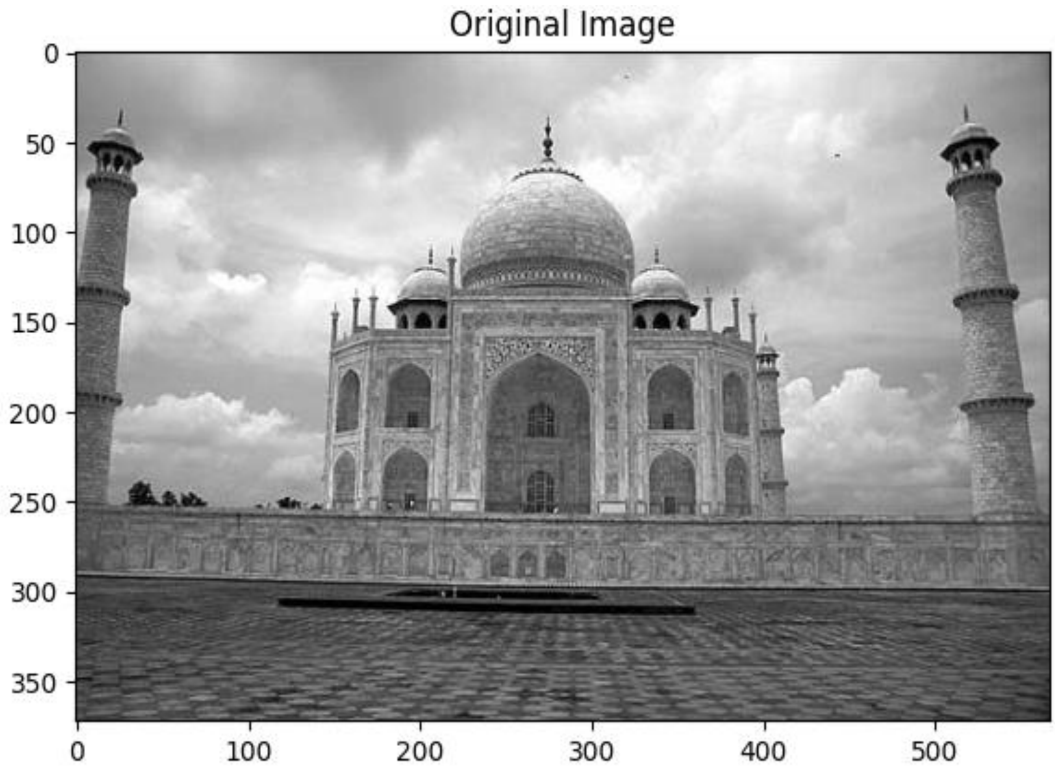
(Course Teacher)

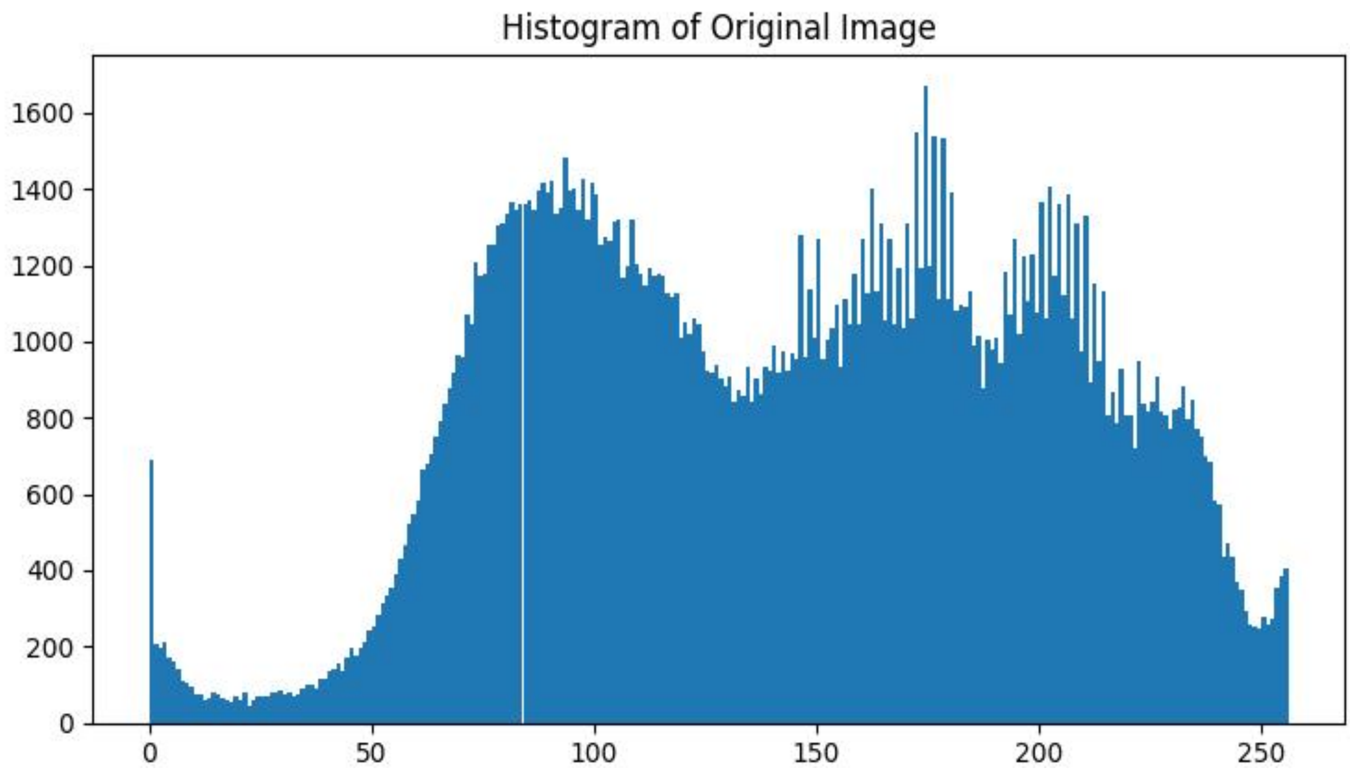
CLASS	**:**	**B.E (E &TC)**	**COURSE**	**:**	**DIVP**
AY	**:**	**2020-21 (SEM- I)**	**DATE**	**:**	**16-10-2020**
EXPT. NO.	**:**	**10**	**CLASS & ROLL NO**	**:**	**BE VIII 42428**
TITLE	**:**	**ENHANCE A GRAY IMAGE USING PSEUDO COLORING**			

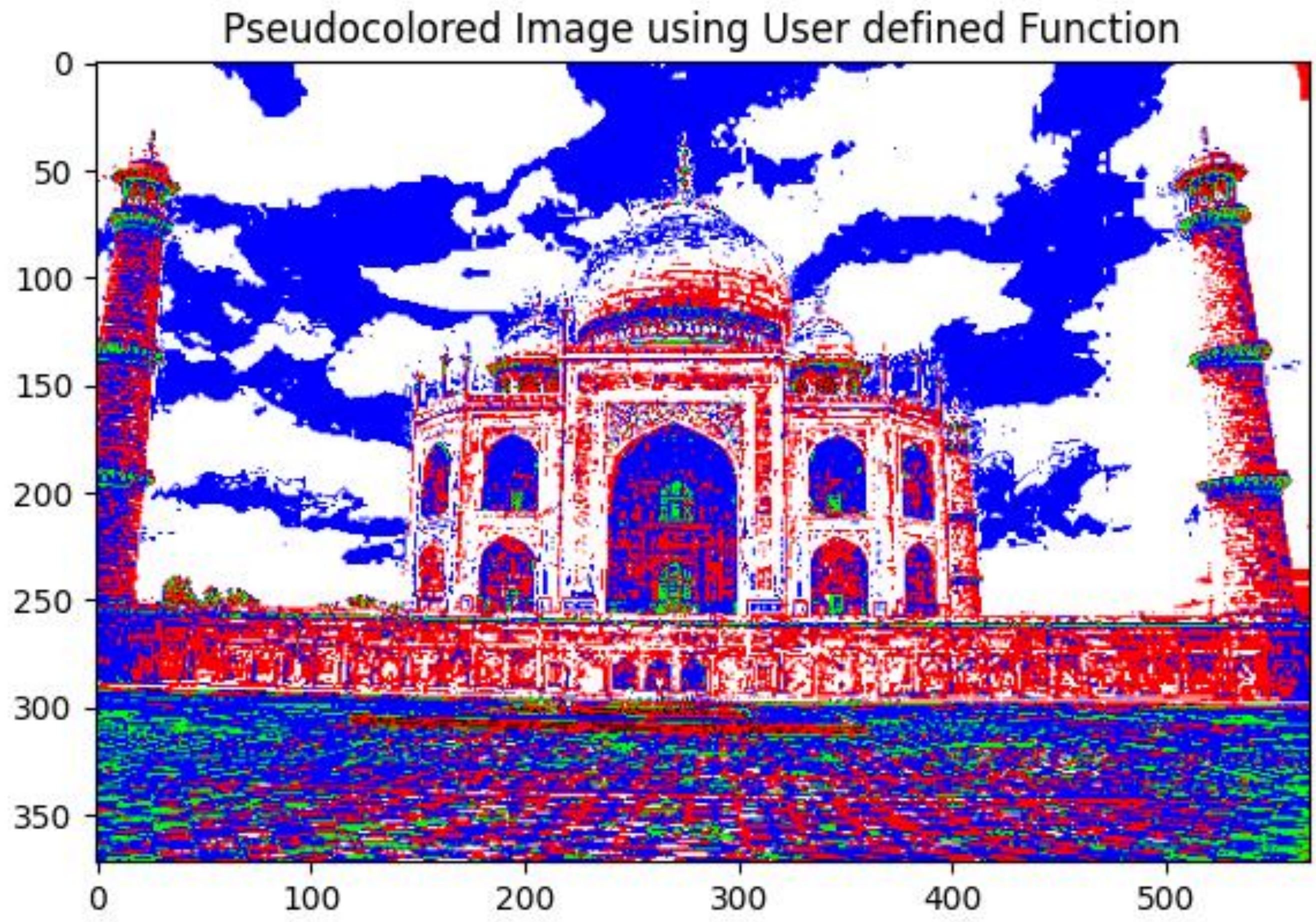
I. CODE:

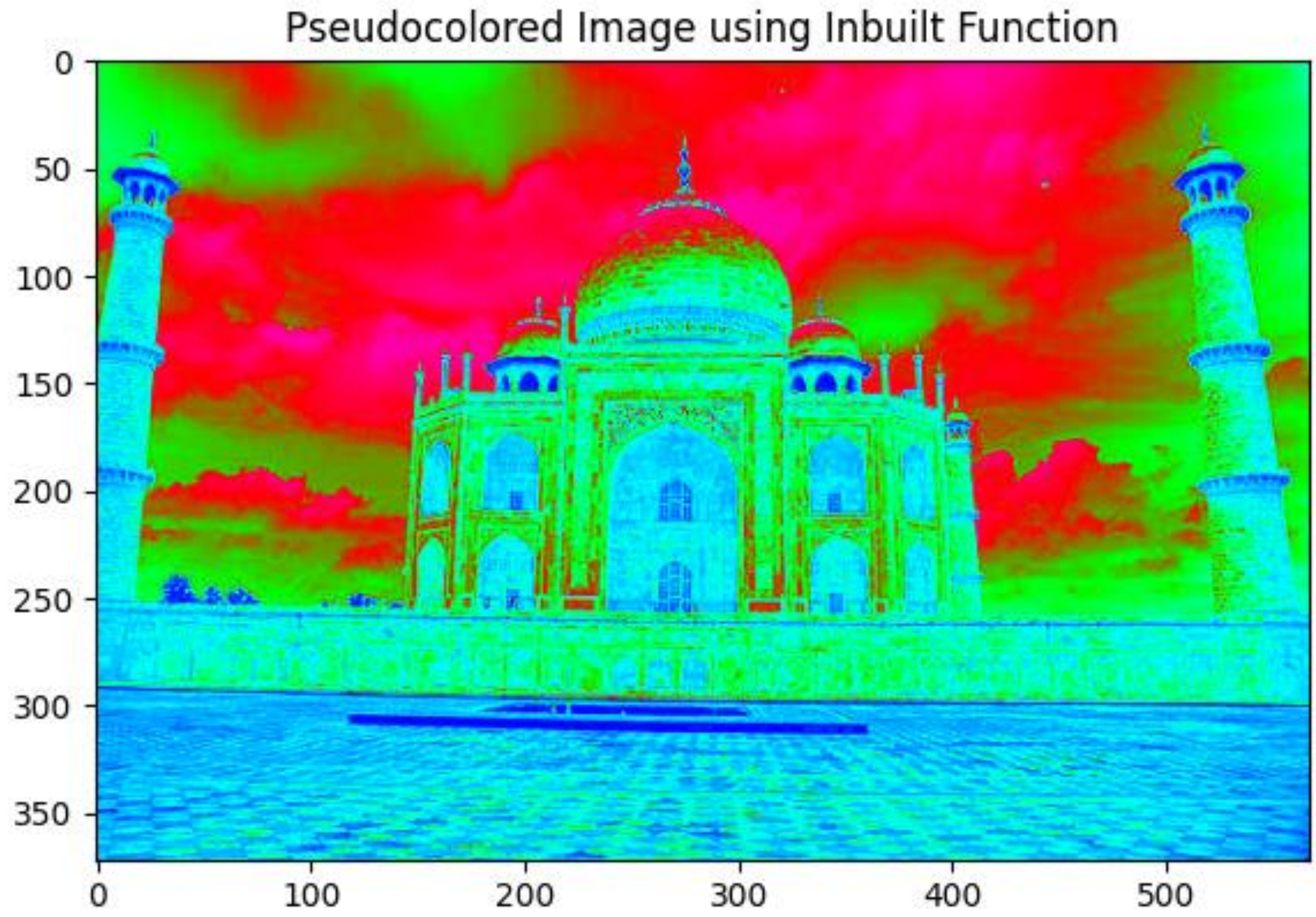
```
import cv2

# import Numpy
import numpy as np
from matplotlib import pyplot as plt

# read a image using imread
img = cv2.imread('Images/tajmahal1.jpg',0)
h,w=img.shape
colored = np.zeros(shape=(h,w,3))

for i in range(0,w):
    for j in range(0,h):
        if(img[j,i]>=0 and img[j,i]<=31):
            colored[j,i,0] = img[j,i];
        elif(img[j,i]>=32 and img[j,i]<=63):
            colored[j,i,1] = img[j,i];
        elif(img[j,i]>=64 and img[j,i]<=95):
            colored[j,i,2] = img[j,i];
        elif(img[j,i]>=96 and img[j,i]<=127):
            colored[j,i,0] = img[j,i]+10;
        elif(img[j,i]>=128 and img[j,i]<=159):
            colored[j,i,0] = img[j,i]-10;
            colored[j,i,1] = img[j,i]-20;
            colored[j,i,2] = img[j,i]+15;
        elif(img[j,i]>=160 and img[j,i]<=191):
            colored[j,i,0] = img[j,i]-10;
            colored[j,i,1] = img[j,i]-10;
            colored[j,i,2] = img[j,i]+15;
        elif(img[j,i]>=192 and img[j,i]<=223):
            colored[j,i,2] = img[j,i]+15;
        else:
            colored[j,i,0] = img[j,i]-10;
            colored[j,i,1] = img[j,i]-20;
            colored[j,i,2] = img[j,i]-30;

im_color = cv2.applyColorMap(img, cv2.COLORMAP_RAINBOW)

plt.subplot(2, 2, 1),plt.imshow(img, 'gray'),plt.title('Original Image')
plt.subplot(2, 2, 2),plt.imshow(colored),plt.title('Pseudocolored Image using User defined Functio
n')
plt.subplot(2, 2, 3), plt.hist(img.ravel(),256,[0,256]),plt.title('Histogram of Original Image')
plt.subplot(2, 2, 4),plt.imshow(im_color),plt.title('Pseudocolored Image using Inbuilt Function')
plt.show()
```

II. RESULTS:


Original Image
0
50
100
150
200
250
300
350
0
100
200
300
400
500


Histogram of Original Image
1600
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250


Pseudocolored Image using User defined Function
0
50
100
150
200
250
300
350
0
100
200
300
400
500


Pseudocolored Image using Inbuilt Function
0
50
100
150
200
250
300
350
0
100
200
300
400
500