

|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name:** | **Digital Image Processing** | **Semester:** | **VII** |
| **Date of Performance:** | **November 26, 2022** | **Batch No:** | **DIP2** |
| **Faculty Name:** | **Prof. Gopal Gupta** | **Roll No:** | **1912052** |
| **Faculty Sign & Date:** |  | **Grade/Marks:** |  |

Experiment No: 8

**Title: To study image compression**

|  |
| --- |
| **Aim and Objective of the Experiment:** |
| To study the process of image compression |

|  |
| --- |
| **COs to be achieved:** |
| **CO1: Understand fundamental of image processing and Computer vision.**  **CO2: Apply enhancement techniques for images in spatial and frequency domain. CO3: Understand and analyze the digital images using different techniques.**  **CO5: Understand the applications in 2D and 3D Vision systems.** |

|  |
| --- |
| **Theory:** |
| What is image compression?  Image compression is a process applied to a graphics file to minimize its size in bytes without degrading image quality below an acceptable threshold. By reducing the file size, more images can be stored in a given amount of disk or memory space. The image also requires less bandwidth when being transmitted over the internet or downloaded from a webpage, reducing network congestion and speeding up content delivery.  What are the two types of image compression?  The methods used to compress image files typically fall into one of two categories: lossy and lossless. Lossy compression reduces an image file size by permanently removing less critical information, particularly redundant data. Lossy compression can significantly reduce file size, but it can also reduce image quality to the point of distortion, especially if the image is overly compressed. However, quality can be maintained when compression is carefully applied.  One of the challenges with lossy compression is that it's irreversible. Once it has been applied to an image, that image can never be restored to its original state. If lossy compression is applied repeatedly to the same image, it gets increasingly distorted. That said, lossy compression has proved to be a valuable strategy for the web, where a moderate amount of image degradation can often be tolerated.  The most common example of lossy compression is JPEG, an image compression format used |

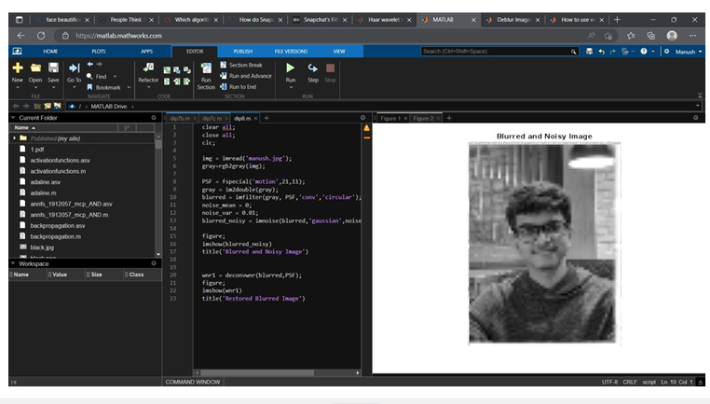


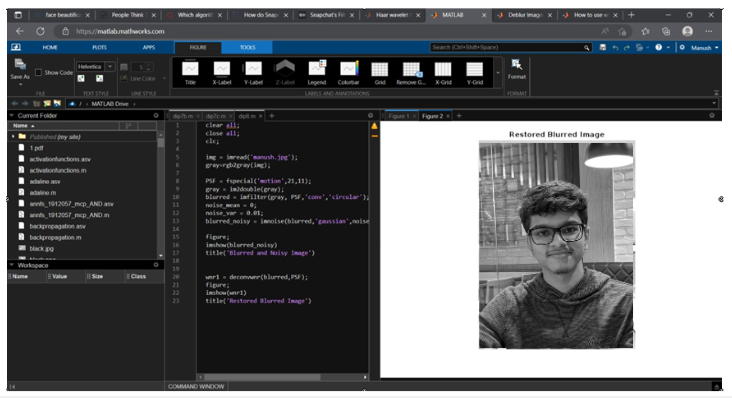
extensively on the web and in digital photography. This widely recognized format is supported by numerous tools and applications. Additionally, compression can be applied in degrees, making it possible to use JPEG compression that best strikes a balance between file size and quality.

|  |
| --- |
| **Stepwise-Procedure:** |
| 1. Take a selfie & Transfer the image PC 2. Add noise with different methods 3. Remove noise using a wiener filter, Band Reject filter, optimum notch filter 4. Compress row image with RLE Huffman, arithmetic & IGS coding 5. Perform segmentation using region grow, split and split &merge 6. Cartoonise yourself 7. Repeat the same steps using Matlab / Python |

|  |
| --- |
| **Output** |
| Upload picture screenshots for all approaches with intermediate steps  **Weiner Filtered Image**  clear all; close all; clc;  img = imread('manush.jpg'); gray=rgb2gray(img);  PSF = fspecial('motion',21,11); gray = im2double(gray);  blurred = imfilter(gray, PSF,'conv','circular'); noise\_mean = 0;  noise\_var = 0.01;  blurred\_noisy = imnoise(blurred,'gaussian',noise\_mean,noise\_var);  figure; imshow(blurred\_noisy)  title('Blurred and Noisy Image')  wnr1 = deconvwnr(blurred,PSF); figure;  imshow(wnr1)  title('Restored Blurred Image') |



****

****

# Huffman Encoding

clear all; close all; clc;



img = imread('vedant.jpg'); gray=rgb2gray(img);

%size of the image [m,n]=size(gray); Totalcount=m\*n;

%variables using to find the probability cnt=1;

sigma=0;

%computing the cumulative probability. for i=0:255

k=gray==i; count(cnt)=sum(k(:))

%pro array is having the probabilities pro(cnt)=count(cnt)/Totalcount; sigma=sigma+pro(cnt); cumpro(cnt)=sigma;

cnt=cnt+1; end;

%Symbols for an image symbols = [0:255];

%Huffman code Dictionary

dict = huffmandict(symbols,pro);

%function which converts array to vector vec\_size = 1;

for p = 1:m for q = 1:n

newvec(vec\_size) = gray(p,q); vec\_size = vec\_size+1;

end end

%Huffman Encodig

hcode = huffmanenco(newvec,dict);

%Huffman Decoding

dhsig1 = huffmandeco(hcode,dict);

%convertign dhsig1 double to dhsig uint8 dhsig = uint8(dhsig1);

%vector to array conversion dec\_row=sqrt(length(dhsig)); dec\_col=dec\_row;

%variables using to convert vector 2 array arr\_row = 1;

arr\_col = 1;

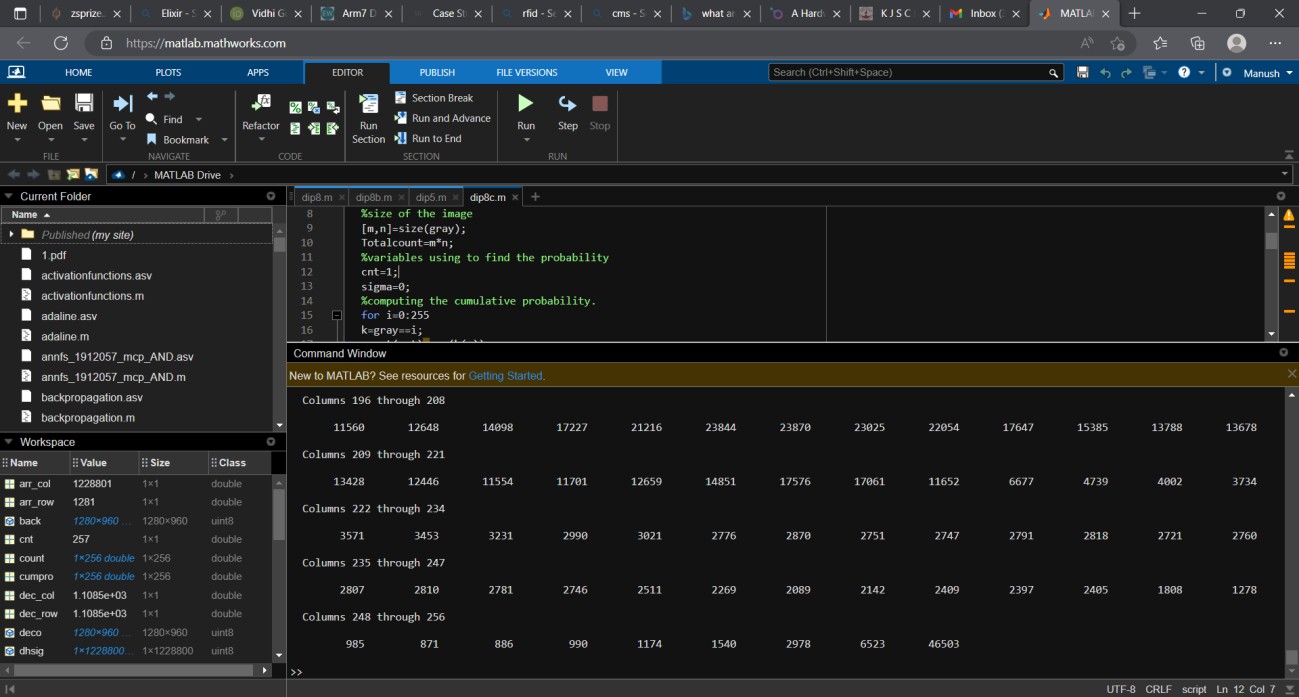
vec\_si = 1; for x = 1:m for y = 1:n

back(x,y)=dhsig(vec\_si); arr\_col = arr\_col+1; vec\_si = vec\_si + 1; end

arr\_row = arr\_row+1; end

%converting image from grayscale to rgb





[deco, map] = gray2ind(back,256); RGB = ind2rgb(deco,map); imwrite(RGB,'decoded.JPG');

%end of the huffman coding

# Cartoon Image

% Convert to HSV color space

% Leave H alone, or quantize it down to some smaller set

% Binary threshold S with a low threshold so pastels push to white

% Binary threshold Vwith a low threshold so dark stuff turns to black

clear all; close all; clc;

img = imread('selfie.JPG'); imgHSV = rgb2hsv(img); [m,n,o] = size(imgHSV);

%new image

newimg = zeros(m,n,o);

figure; imshow(img);

title('Original image');

for i=1:m

for j=1:n



for k=1:o

newimg(i,j,k) = imgHSV(i,j,k);

if (imgHSV(i,j,1)>0.3)

newimg(i,j,1) = 1;

end

if (imgHSV(i,j,2)>0.2)

newimg(i,j,2) = 1;

end

if (imgHSV(i,j,3)>0.4)

newimg(i,j,3) = 1;

end

end

end

end

% newimg = int8(newimg); figure;

imshow(newimg);

title('HSV adjusted image image');

% Opened image

se = strel("line", 15, 15); open = imopen(newimg, se);

myCartoon = hsv2rgb(open); figure; imshow(myCartoon);

title('final cartoon image'); imwrite(myCartoon,"mycartoon2.png")



|  |
| --- |
| **Conclusions:** |
| In this experiment we added noise to an image that is selfie and used Wiener and notch filtering to remove noise. We also implemented image compression techniques. We also tried cartoonifying selfie. |



|  |
| --- |
| **Post Lab Subjective/Objective type Questions:** |
| Answer the following questions:  1. What are the different image compression methods  **ANS**: There are two types of image compression techniques – Lossy and Lossless. Lossless compression is a technique used to reduce the file size of an image while maintaining its quality like before. The lossy compression technique is another way of image compression that involves cutting some part of the image to create even smaller file sizes. When practiced in the right way and combined with dithering, this can lead to images that are almost identical to the original images.  **Lossless Image Compression Techniques:**   * Run-length Encoding * Predictive Coding and DPCM * Area Image Compression * DEFLATE * Chain Codes * Adaptive Dictionary Algorithms * Entropy Encoding   **Lossy Image Compression Techniques:**   * Decreasing the Color Space * Transform coding * Chroma Subsampling * Fractal Compression |

**Signature of faculty in-charge with Date:**