

BSc(H) Computer Science (Semester – V)
DSE: Digital Image Processing
Assignment – 1

Name – Prerna

College Roll no. – 20201420

Exam Roll no. – 20020570025

Q1) Implement the JPEG compression algorithm. JPEG uses the DCT Transform. Create three versions of the algorithm: 1) Using the standard DCT Transform, 2) Using Fourier Transform, 3) Using Wavelet Transform.

Compare the performance of the three versions after application of the compression scheme on an image. The comparison will be on the basis of the standard compression performance indices used. (Paste your code and output in the file to be submitted)

#SOLUTION

1(a) Standard DCT Algorithm :-

The discrete cosine transform (DCT) is a technique for converting a signal into elementary frequency components. Like other transforms, the Discrete Cosine Transform (DCT) attempts to de correlate the image data. After de correlation each transform coefficient can be encoded independently without losing compression efficiency.

DCT Algorithm:

- The following is a general overview of the JPEG process.
- The image is broken into 8x8 blocks of pixels.
- Working from left to right, top to bottom, the DCT is applied to each block.
- Each block is compressed through quantization.
- The array of compressed blocks that constitute the image is stored in a drastically reduced amount of space.
- When desired, the image is reconstructed through decompression, a process that uses the inverse Discrete Cosine Transform (IDCT)

#CODE

```
I = imread('peppers.png');  
Y_d = rgb2ycbcr(I);  
// Downsample:  
Y_d(:, :, 2) = 2*round(Y_d(:, :, 2)/2);  
Y_d(:, :, 3) = 2*round(Y_d(:, :, 3)/2);
```

```

// DCT compress:
A = zeros(size(Y_d));
B = A;
for channel = 1:3
    for j = 1:8:size(Y_d,1)-7
        for k = 1:8:size(Y_d,2)
            II = Y_d(j:j+7,k:k+7,channel);
            freq = chebfun.dct(chebfun.dct(II).').';
            freq = Q.*round(freq./Q);
            A(j:j+7,k:k+7,channel) = freq;
            B(j:j+7,k:k+7,channel) = chebfun.idct(chebfun.idct(freq).').';
        end
    end
end

subplot(1,2,1)
imshow(ycbcr2rgb(Y_d))
title('Original')
subplot(1,2,2)
imshow(ycbcr2rgb(uint8(B)));
title('Compressed')

```



1(b)

The Discrete Fourier transform (DFT) is applied to each $M \times N$ block independently to represent the image in the frequency domain yielding the real and imaginary components. The Matrix Minimization algorithm is applied to each component and zeros are removed. The resulting vectors are subjected to Arithmetic coding and represent the compressed data.

A uniform quantization is then applied to both parts, which involves dividing each element by a factor called quantization factor Q followed by rounding the outcomes which results in an increase of high frequency coefficients probability thus reducing the number of bits needed to represent such coefficients. The result of this operation is that the compression ratio increases.

#CODE

```

a = imread('Lenna.jpg');
[m,n] = size(a);
imshow(a,"Original Image");
A = fft2(double(a));
figure(1)
imshow(abs(A),'2D discrete transform image');

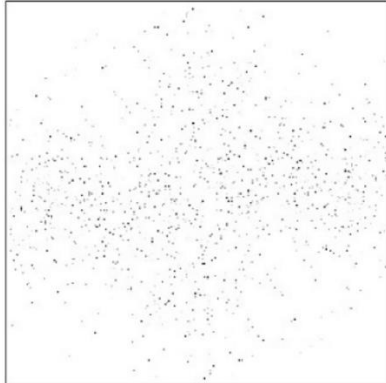
```

```

B = fftshift(A)
figure(2)
imshow(abs(B),'fft shifted' );
a_inv = fft2 (A')
a_inv = a_inv'/( m * n );
figure(3)
imshow( uint8(abs( a_inv )), 'IDFT' );

```

2D-Discrete Fourier Transform of Lena Image



1(c)

The wavelet analysis method is a time-frequency analysis method which selects the appropriate frequency band adaptively based on the characteristics of the signal. Then the frequency band matches the spectrum which improves the time-frequency resolution.

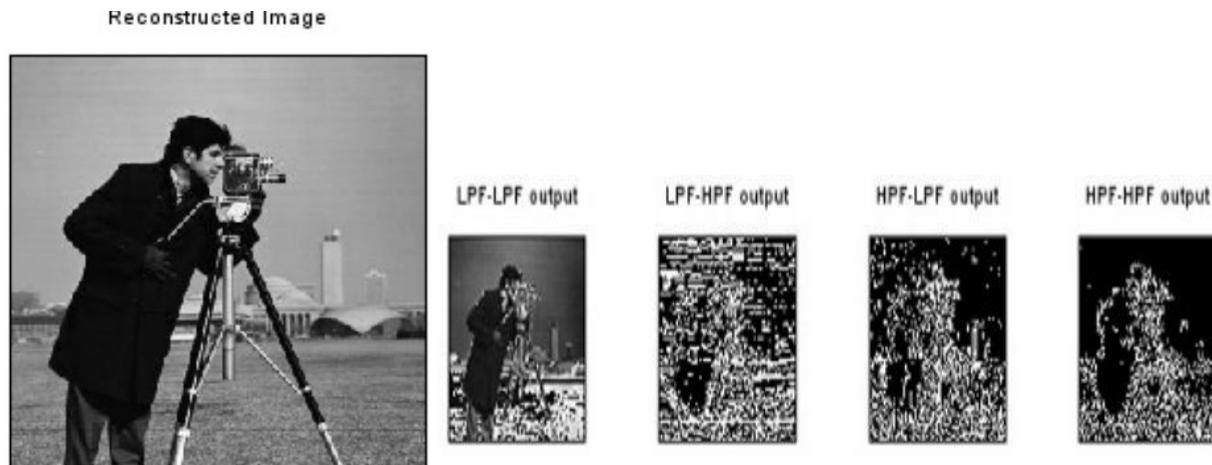
#CODE

```

rgb = imread("cameraman.jpg")
[LL,LH,HL,HH] = dwt2(rgb, "haar")
subplot(2,2,1);imshow(LL/255);title('LL band of image');
subplot(2,2,2);imshow(LH);title('LH band of image');
subplot(2,2,3);imshow(HL);title('HL band of image');
subplot(2,2,4);imshow(HH);title('HH band of image');
figure
subplot(1,2,1)
imshow(rgb)
title('Original Image')

subplot(1,2,2)
imshow('peppers.jpeg')
title('JPEG compressed')

```



Q2) Refer to the paper Sarma, Rituparna, and Yogesh Kumar Gupta. "A comparative study of new and existing segmentation techniques." *IOP Conference Series: Materials Science and Engineering*. Vol. 1022. No. 1. IOP Publishing, 2021. It presents a number of image segmentation techniques; select any of these and describe the working of the algorithm. Implement and apply the algorithm on an image. (Paste your code and output in the file to be submitted)

#SOLUTION

Edge detection is more common for detecting discontinuities in gray level than detecting isolated points and thin lines because isolated points and thin lines so not occur frequently in most practical images. The edge is the boundary between two regions with relatively distinct gray level properties. It is assumed here that the transition between two regions can be determined on the basis of gray level discontinuities alone.

A. Sobel Operators

The computation of the partial derivation in gradient may be approximated in digital images by using the Sobel operators which are shown in the masks below:

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	1

These two masks together with any of the equations:

$$|\nabla f| = \sqrt{G_x^2 + G_y^2}$$

$$|\nabla f| = |G_x| + |G_y|$$

are used to obtain the gradient magnitude of the image from the original.

B. Roberts Cross Edge Detector

The Roberts Cross operator performs a simple, quick to compute, 2-D spatial gradient measurement on an image. It thus highlights regions of high spatial frequency which often correspond to edges. In its most common usage, the input to the operator is a grayscale image, as is the output. Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient of the input image at that point.

1	0
0	-1

0	+1
-1	0

D. Prewitt Operator

The prewitt operator uses the same equations as the Sobel operator, except that the constant $c = 1$. Therefore:

Note that unlike the Sobel operator, this operator does not place any emphasis on pixels that are closer to the centre of the masks. The Prewitt operator measures two components. The vertical edge component is calculated with kernel G_x and the horizontal edge component is calculated with kernel G_y . $|G_x| + |G_y|$ give an indication of the intensity of the gradient in the current pixel.

-1	0	1
-1	0	1
-1	0	1

G_x

1	1	1
0	0	0
-1	-1	-1

G_y

#CODE

```
Img = imread('tru.jpg');
Img = im2gray(Img);
imshow(Img);
e = edge(Img); // sobel, thresh = 0.5
imshow(e,2)
```

```
e = edge(Img, 'prewitt'); // thresh = 0.5
imshow(e,2)
e = edge(Img, 'Roberts', [0.06 0.2]);
imshow(e,2)
```



Q3) Refer to any recent research publication on low pass/high pass filtering on images. Describe the algorithm used. Present an analysis of the same and enumerate its strength and weaknesses. Extra points for implementing the algorithm used. Papers can be found at Google Scholar.

#Solution

FILTERING TECHNIQUES

The major three filtering techniques that have been opted for analysis are the high pass, low pass, and median filtering techniques.

#LOW PASS FILTERS

The low pass filters are majorly used for removing the high-frequency noise content from real-time images. This involves smoothing of sharp edges. The noise content can be of Gaussian type, or uniform or Rayleigh or else in nature. This filter can be further categorized as Low Pass Averaging Filter and Low Pass Median Filter. These filter techniques use the masking techniques which blur the sharp edges of the image under consideration. When the mask is applied to the image, it reduces the sudden transition of the pixel values resulting in a reduction of sharp edges.

1	1	1
1	1	1
1	1	1

The above image shows a typical 3*3 matrix mask used for low filtering techniques. As shown, the sum of elements of the matrix equals to one. In order to implement the mask, the image undergoes the following mathematical expression, where $w(n)$, such that $n=1, 2, 3...9$ corresponds to the corresponding elements of the masking matrix.

$$a1(x, y) = w(1) * a(x - 1, y - 1) + w(2) * a(x - 1, y) + w(3) * a(x - 1, y + 1) + w(4) * a(x, y - 1) + w(5) * a(x, y) + w(6) * a(x, y + 1) + w(7) * a(x + 1, y - 1) + w(8) * a(x + 1, y) + w(9) * a(x + 1, y + 1) \quad (1)$$

The above instruction is to be implemented in a loop with an initialization of $x=2$ and $y=2$. This results in determining the value of the anchor element $([2, 2],$ in this case) which will decide the corresponding pixel value of the new filtered image. The below example of an image pixel set determines how the average low pass filter mask causes reduction the sharpness of the image by reducing the shift in pixel values.

60	60	65	65	65
15	15	15	20	25
20	20	20	20	30
80	80	80	90	90
95	90	90	95	95

As the arrow shows, there has been a sudden shift in the pixel value as we move from one row to the other. Now applying the average filter mask as shown earlier (Consider the shaded region). Applying the mathematical expression and determining the anchor values after each loop execution, we get the new pixel values of the shaded portion as follows.

15	15	15
20	20	20
80	80	90

↓

31.66	32.78	35.55
38.33	40	43.33
63.88	65	67.77

As we can observe from the image, the sudden increase of 300% in case of transition from 20 to 80 and an increase of 350% with transition from 20 to 90, whereas, with the application of mask, the transition is occurring at a rate of 166% and 156% resulting in reduction of pixel change and making the image less sharp and smoother.

#HIGH PASS FILTERS

Coming to the high pass filters, they retain the high-frequency contents and are majorly used in the sharpening of the images. A typical example of high pass filter mask is as follows.

0	-4	0
-4	16	-4
0	-4	0

The major limitation of high pass filters is that they remove the background to a large extent. Thus, if we require to keep the background along with the enhancement of the image, we may scale the mask elements such that the net sum is not zero. The median filter is similar to low pass filter used majorly for reduction in noise. The methodology involves the arrangement of the mask in a single row matrix in a sorted manner. The sorting can be performed in both ascending as well in descending order. The medium of the sorted value is then taken as the anchor value of that element. This process ensures that the obtained value will always remain within the pixel range which was not possible in case of average low pass filter. The latter part introduces the analysis process of the three filtering techniques.

#CODE

#Spatial High Pass

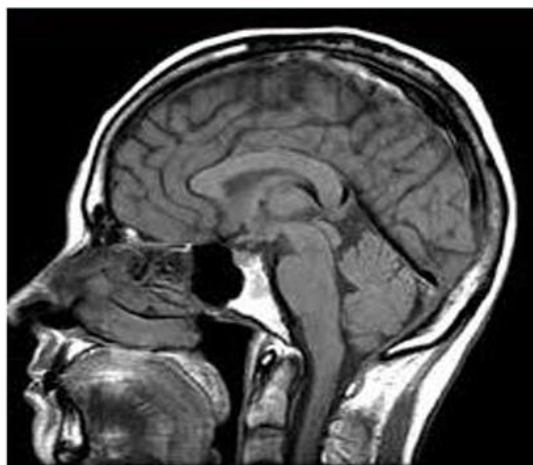
```
i1 = imread('Test_images/brain_tumor.jpg');
g_filter = fspecial('gaussian');
i2 = imfilter(i1, g_filter);
g_filter2 = fspecial('gaussian', [8,8], 10);
i3 = imfilter(i1, g_filter2);
g_filter3 = fspecial('gaussian', [25,25], 31);
i4 = imfilter(i1, g_filter3);
```

#Spatial Low Pass

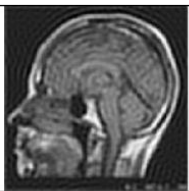


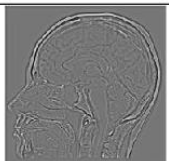
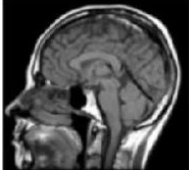
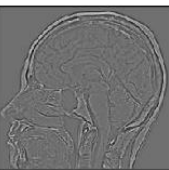
```
/* Spatial Low Pass Filter */
i1 = imread('Test_images/lena.jpeg');
l_filter = fspecial('laplacian');
i2 = imfilter(i1, l_filter);
```

#Butter Worth

```
/* b. low & high pass filtering */
// Butterworth Filters
G11 = mkfftfilter(img, 'butterworth1', 0.3);
H11 = 1 - G11;
subplot(121), title('DFT Butterworth Low Pass Image'), imshow(G11);
subplot(122), title('DFT Butterworth High Pass Image'), imshow(H11);
```

Original Image

Name of the low pass filter	Resultant Image	Name of the high pass filter	Resultant Image
Ideal		Ideal	
Butterworth		Butterworth	
Gaussian		Gaussian	

REFERENCES:-

- [1] Amit Shukla, Dr R.K. Singh, "Performance Analysis of Frequency Domain Filters For Noise Reduction", e-Journal of Science & Technology (e-JST).
- [2] AyushDogra and ParvinderBhalla, "Image Sharpening By Gaussian And Butterworth High Pass Filter", Biomedical & Pharmacology Journal, Vol. 7(2), 707-713 2014.
- [3] Aziz Makandar& Bhagirathi Halalli, "Image Enhancement Techniques using High pass and Low pass Filters", International Journal of Computer Applications (0975 – 8887), Vol. 109 – No. 14, January 2015.

