LAB Manual

PART A

(PART A : TO BE REFFERED BY STUDENTS)

**Experiment No.05**

**A.1 Aim:**

Write a program to apply various (Hadamard, Walsh and DFT) transforms on an image and compare the results.

**A.2 Prerequisite:**

1 Matlab programming syntax (Refer the Matlab manual).

2. Knowledge Hadamard, Walsh and Fast Fourier Transform.

2. Availability of Soft copy of your 3 Photographs with different background (i.e. Plane, scenery,

etc.) for experiment.

**A.3 Outcome:**

**After successful completion of this experiment students will be able to**

1. Understand the fundamentals of Hadamard, Walsh and DFT Transforms and its effects on digital images.
2. Appreciate different properties of the Hadamard, Walsh and DFT transform.
3. Apply and verify the correctness of Hadamard, Walsh and DFT tranfomrs on images.
4. Identify applications of transforms studied.

**A.4 Theory:**

**Hadamard Transform**

The Hadamard transform is based on the Hadamard matrix which is a square array

having entries of +1 or -1 only. The Hadamard matrix of order 2 is given by

**H2 =**

The rows and columns are orthogonal. For orthogonality of vectors the dot product

has to be zero. We get H(4) from the Kronecker product of H(2)

H(4) = H(2) X H(2)

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1 | 1 | 1 |
| 1 | -1 | 1 | -1 |
| 1 | 1 | -1 | -1 |
| 1 | -1 | -1 | 1 |

H4 =

So we know that the Hadamard matrices of order 2n can be recursively generated

H(2n) = H(2) X H(2n-1)

The rows of Hadamard matrix can be considered to be samples of rectangular

waves with sub-periods of 1/N units.

If x(n) is N-point 1 dimensional sequence of finite valued real numbers arranged in

a column then the Hadamard transformed sequence is given by

X = T.x X[n] = [H(N) x(n)]

The inverse Hadamard transform is given by

x(n) = 1/N H(N) X(n)

For a two dimensional sequence f of size N X N, we compute the Hadamard

transform using equation

F = T f T F = [H(N) f H(N)]

The inverse Hadamard transform is given by,

f = TFT f = [H(N) F H(N)]

**Walsh Transform**

The Walsh matrix was proposed by Joseph Leonard Walsh in 1923. The Walsh matrix is formed rearranging the rows of Hadamard Matrix so that the number of sign-changes(sequency) in a row is in increasing order. The Walsh matrix of size 4 is as given below

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | Sequency |
| 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | -1 | -1 | 1 |
| 1 | -1 | -1 | 1 | 2 |
| 1 | -1 | 1 | -1 | 3 |

W4 =

where the successive rows have 0, 1, 2 and 3 sign changes.

If x(n) is N-point 1 dimensional sequence of finite valued real numbers arranged in

a column then the Walsh transformed sequence is given by

X = T.x X[n] = [W(N) x(n)]

The inverse Walsh transform is given by

x(n) = 1/N W(N) X(n)

For a two dimensional sequence f of size N X N, we compute the Walsh

transform using equation

F = T f T F = [W(N) f W(N)]

The inverse Hadamard transform is given by,

f = TFT f = [W(N) F W(N)]

**Discrete Fourier Transform**

In spatial domain, we perform convolution of filter mask with image data. In frequency domain we perform multiplication of Fourier transform of image data with filter transfer function.

The general idea is that the image (***f(x,y)*** of size ***M***x***N***) will be represented in the frequency domain (***F(u,v)***). The equation for the two-dimensional discrete Fourier transform (DFT) is:

DFT equation

The concept behind the Fourier transform is that any waveform can be constructed using a sum of sine and cosine waves of different frequencies. The exponential in the above formula can be expanded into sines and cosines with the variables ***u*** and ***v*** determining these frequencies.

The inverse of the above discrete Fourier transform is given by the following equation:

Inverst DFT equation

Thus, if we have ***F(u,v)***, we can obtain the corresponding image (***f(x,y)***) using the inverse, discrete Fourier transform.

Things to note about the discrete Fourier transform are the following:

* the value of the transform at the origin of the frequency domain, at ***F(0,0)***, is called the dc component
  + ***F(0,0)*** is equal to ***MN*** times the average value of ***f(x,y)***
  + in MATLAB, ***F(0,0)*** is actually ***F(1,1)*** because array indices in MATLAB start at 1 rather than 0
* the values of the Fourier transform are complex, meaning they have real and imaginary parts. The imaginary parts are represented by i, which is defined solely by the property that its square is −1, ie:http://www.cs.uregina.ca/Links/class-info/425/Lab5/Equations/imaginary_definition.png
* we visually analyze a Fourier transform by computing a **Fourier spectrum** (the magnitude of ***F(u,v)***) and display it as an image.
  + the Fourier spectrum is symmetric about the origin
* the fast Fourier transform (FFT) is a fast algorithm for computing the discrete Fourier transform.
* MATLAB has three functions to compute the DFT:
  + fft -for one dimension (useful for audio)
  + fft2 -for two dimensions (useful for images)
  + fftn -for n dimensions
* MATLAB has three related functions that compute the inverse DFT:
  + idft
  + idft2

**A.5 Procedure/Algorithm:**

**A.5.1:**

**TASK 1:**

1. Read the i/p image

2. Resize the image to convert it into square matrix

3. Generate Hadamard and Walsh transform matrices of size equivalent to the size

of image.

4. Apply Hadamrd and Walsh Transform on the image separately.

5. Display the transformed images w.r.t. particular transform applied.

6. Regenerate and display the original image back

7. Compare the input and output images for each transforms applied w.r.t. its

matrix content and visibility on the screen.

8. Add the original image with the transformed output image of each

Transformation function applied separately and observe the result.

9. Save and close the file and name it as **EX5\_Task1\_your Roll no.m**

**A.5.2:**

**TASK 2:**

1. Read the i/p image

2. Resize the image to convert it into square matrix

3. Transform the image using DFT.

4. Verify and note the matrix content of the transformed image in workspace.

5. Display the transformed image.

6. Display the magnitude and phase images out of the transformed image

7. Label each output appropriately.

8. Regenerate and display the original image back

9. Compare the input and output images w.r.t. its matrix content and visibility

on the screen

10. Add the original image with the transformed output image (before applying

inverse transform on it) separately (for all 3 outputs) and observe the result.

11. Save and close the file and name it as **EX5\_Task2\_your Roll no.m**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

***(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)***

|  |  |
| --- | --- |
| Roll No : B032 | Name: Naman Garg |
| Class : B.Tech CS-B | Batch : B2 |
| Date of Experiment: 26.8.2020 | Date of Submission |
| Grade : | Time of Submission: |
| Date of Grading: |  |

**B.1 Software Code written by student:**

img=imread('gg.jpg');

img=rgb2gray(img);

img=imresize(img,[256,256],'nearest');

h=hadamard(256);

h\_t=h.';

F=h\*double(img)\*h\_t/256;

f=h\_t\*F\*h/256;

F(1,1)=0;

disp(isequal(F,zeros([256,256])));

res=fft2(img);

res\_inv=ifft2(res);

c=zeros(256,1);

for i=1:256

sign\_change=0;

for j=1:255

if(h(i,j)\*h(i,j+1)==-1)

sign\_change=sign\_change+1;

end

end

c(i)=sign\_change;

end

w=zeros(256,256);

for i=1:256

w(i,:)=h(find(c==i-1),:);

end

w\_t=w.';

G=w\*double(img)\*w\_t/256;

g=w\_t\*G\*w/256;

subplot(3,5,1),imshow(uint8(real(res))),title('FFT-Real');

subplot(3,5,2),imshow(uint8(imag(res))),title('FFT-Imag');

subplot(3,5,3),imshow(uint8(atand(imag(res)/real(res)))),title('FFT-Angle');

subplot(3,5,4),imshow(uint8(abs(res))),title('FFT-Magnitude');

subplot(3,5,5),imshow(uint8(res\_inv)),title('Inverse');

subplot(3,5,6),imshow(img),title('INPUT');

subplot(3,5,7),imshow(uint8(h)),title('h Matrix');

subplot(3,5,8),imshow(uint8(F)),title('h Transform');

subplot(3,5,9),imshow(uint8(f)),title('Inv h Transform');

subplot(3,5,11),imshow(img),title('INPUT');

subplot(3,5,12),imshow(uint8(w)),title('w Matrix');

subplot(3,5,13),imshow(uint8(G)),title('w Transform');

subplot(3,5,14),imshow(uint8(g)),title('Inv w Transform');

**B.2 Input and Output:**

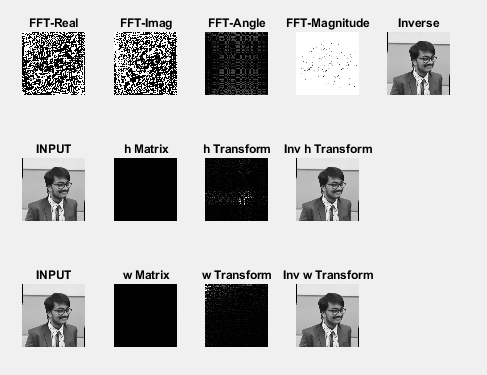
**Input Images:**

** **

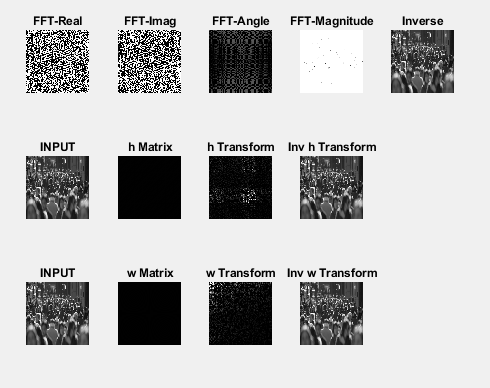
**Output Images:**

1. **For each Transform (Hadamard, Walsh and DFT ) as per the procedure discussed in section A.5.**

**Naman’s Image:**

****

**Crowd Image:**

****

**B.3 Observations and learning:**

***(Students are expected to comment on the output obtained with clear observations and learning for each task/ sub part assigned)***

I learnt about the image processing transformations in the frequency domain. These transformations were Hadamard transform, Fast Fourier Transform and Walsh Transform. I successfully implemented these techniques on 2 photos ie my photo and a crowd photo and observed the results.

Hadamard matrix is a symmetric matrix consisting of only 1 and -1. A hadamard matrix of shape 2^n can be generated by recursively multiplying the matrix with the elements of the matrix. The hadamard matrix when transformed on an image the output pixels will be the sum and difference of rows, columns and diagonals ie permutation of plus and minus operations between the rows,columns and diagonals. For example : A 2x2 hadamard transformation on a 2x2 image will give a 2x2 output where the first value is the sum of all pixels, second value is the difference between column 1 and column 2, third value is the difference between row 1 and row 2 and the last value is the difference between the 2 diagonals. This is why a plain background containing only 1 pixel intensity in all pixels will give a pitch black output with only the top left pixel being white. An image with more intensity changes ie an image with more details (the crowd image) will give more white pixels as the minus operations wont cause those pixels to go entirely black. I got my original image back by doing the inverse hadamard transform

Fast Fourier transform is based on the principle that a signal is composed of multiple sinusoidal waves with different amplitudes and frequencies. We can consider the amplitude as the magnitude of how much a single wave with a particular frequency contributes to the overall wave. FFT finds its application in image processing in the frequency domain.

As we can see from the workspace for the variables, the values in the res matrix (output of FFT) are complex numbers. The complex numbers have very high positive and very low negative values as the real and imaginary parts example : -1.4x 10^5 + 8.1x10^4 i. The negative values get clipped at 0 and the positive values at 255. Hence we see a distortion sort of an image of black and white pixels in the real and imaginary matrix images. However while calculating the magnitude matrix ( magnitude = root of (square of real + square of imaginary)) ,due to the squaring of terms, the sign gets removed and we get high positive magnitude values which get clipped at 255 leading to an overall white image. However it does have some black pixels which could be because of very low values of both real and imaginary parts for those pixels. The less detailed image of my face had an overall darker FFT output while the more detailed crowd image had a brighter FFT output ie a higher density of whiter pixels in the output. I got my original image back by doing inverse fast fourier transform.

Walsh transform is applying sequencing on the Hadamard matrix based upon the sign changes in the row. Hence the ordering of pixels row wise change when compared to Hadamard transform. Also we can observe that the white pixels of the image seem to be concentrated at the top left. Also the same fact applies here that the number of white pixels in the transformed output is directly proportional to the amount of details in the input image. This difference can be seen in my crowd image where the amount of white pixels is comparatively high. I got my original image back by doing the inverse of walsh transform.

**B.4 Conclusion:**

*(****Students must write the conclusion as per the attainment of individual outcome listed above and learning/observation noted in section B.3)***

I learnt and successfully implemented the transformation techniques in the frequency domain ie. Hadamard transform , Fast Fourier Transform and Walsh Transform . I observed the outputs in both my image where there were less details and in the crowd image where there were more details. The details are represented by white pixels so the abundance of these pixels can suggest us the level of details in the image. I verified these transforms by calculating the inverse of them and getting my original image back. I understood the applications of each of these transforms.

**B.5 Question of Curiosity**

***(To be answered by student based on the practical performed and learning/observations)***

Q1: What are the applications [apart from given in Q2.] of each of these transforms you have studied?

1.Hadamard

The Hadamard transform is also used in data encryption, as well as many signal processing and data compression algorithms, such as JPEG XR and MPEG-4 AVC. In video compression applications, it is usually used in the form of the sum of absolute transformed differences. It is also a crucial part of Grover's algorithm and Shor's algorithm in quantum computing. The Hadamard transform is also applied in experimental techniques such as NMR, mass spectrometry and crystallography. It is additionally used in some versions of locality-sensitive hashing, to obtain pseudo-random matrix rotations.

2.Walsh

Applications of the Walsh functions can be found wherever digit representations are used, including speech recognition, medical and biological image processing, and digital holography.For example, the fast Walsh–Hadamard transform (FWHT) may be used in the analysis of digital quasi-Monte Carlo methods.

In radio astronomy, Walsh functions can help reduce the effects of electrical crosstalk between antenna signals. They are also used in passive LCD panels as X and Y binary driving waveforms where the autocorrelation between X and Y can be made minimal for pixels that are off.

3.Dft

The twentieth century has seen the extension of these methods to all linear partial differential equations with polynomial coefficients, and by extending the notion of Fourier transformation to include Fourier integral operators, some non-linear equations as well. Perhaps the most important use of the Fourier transformation is to solve partial differential equations. Many of the equations of the mathematical physics of the nineteenth century can be treated this way

Q2: How can DCT be used for data compression and Steganography?

The secret message is encrypted first then LSB technique is applied. Moreover, DCT is used to transform the image into the frequency domain. The LSB algorithm is implemented in spatial domain in which the payload bits are inserted into the least significant bits of cover image to develop the stego-image while DCT algorithm is implemented in frequency domain in which the stego-image is transformed from spatial domain to the frequency domain and the payload bits are inserted into the frequency components of the cover image. The performance of this technique is evaluated on the basis of the parameters MSE and PSNR.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*