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.: N049 | Name: Tarun Tanmay |
| Class: MBATech CE | Batch: B3 |
| Date of Experiment: | Date of Submission |
| Grade : | Time of Submission: |
| Date of Grading: |  |

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

clear all;

clc;

a=imread("/Users/tjrox0825/Desktop/bw.jpg");

ag=rgb2gray(a);

ar=imresize(ag,[128,128]);

ad=double(ar);

n=128\*128;

%Hadamard

h2=[1 1; 1 -1];

h4=kron(h2,h2);

h8=kron(h2,h4);

h16=kron(h2,h8);

h32=kron(h2,h16);

h64=kron(h2,h32);

h128=kron(h2,h64);

htimg=h128\*ad\*h128;

invht=(h128\*htimg\*h128)/n;

figure('name','Figures');

subplot(3,3,1);

imshow(ar);

title('Original image');

ht=uint8(htimg);

subplot(3,3,2);

imshow(ht);

title('Hadamard image');

iht=uint8(invht);

subplot(3,3,3);

imshow(iht);

title('Inverse image');

%DFT

dftimg=fft2(ad);

inv\_dftimg=ifft2(dftimg);

subplot(3,3,4);

imshow(ar);

title('Original image');

df=uint8(dftimg);

subplot(3,3,5);

imshow(df);

title('Discrete Fourier image');

idf=uint8(inv\_dftimg);

subplot(3,3,6);

imshow(idf);

title('Inverse image');

%Walsh

w8=[];

c=[];

for i=1:8

c(i)=0;

for j=2:8

if (h8(i,j)>0 && h8(i,j-1)<0) || (h8(i,j)<0 && h8(i,j-1)>0)

c(i)=c(i)+1;

end

end

end

for i=1:8

m=find(c==min(c));

for j=1:8

w8(i,j)=h8(m,j);

end

c(m)=9;

end

disp("Hadamard 8x8:");

disp(h8);

disp("Walsh 8x8:");

disp(w8);

w128=[];

c=[];

for i=1:128

c(i)=0;

for j=2:128

if (h128(i,j)>0 && h128(i,j-1)<0) || (h128(i,j)<0 && h128(i,j-1)>0)

c(i)=c(i)+1;

end

end

end

for i=1:128

m=find(c==min(c));

for j=1:128

w128(i,j)=h128(m,j);

end

c(m)=129;

end

wtimg=w128\*ad\*w128;

invwt=(w128\*wtimg\*w128)/n;

subplot(3,3,7);

imshow(ar);

title('Original image');

wt=uint8(wtimg);

subplot(3,3,8);

imshow(wt);

title('Walsh image');

iwt=uint8(invwt);

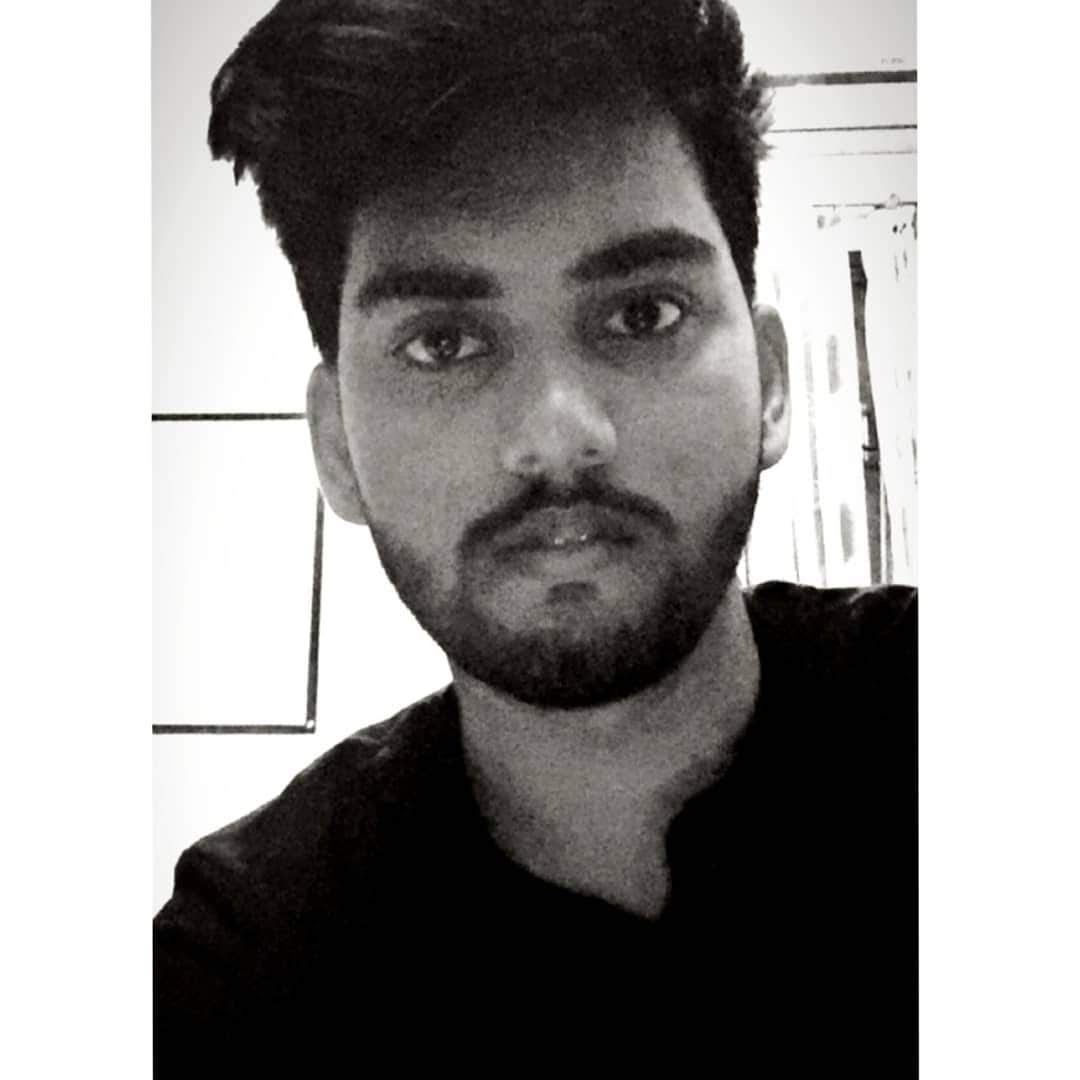
subplot(3,3,9);

imshow(iwt);

title('Inverse image');

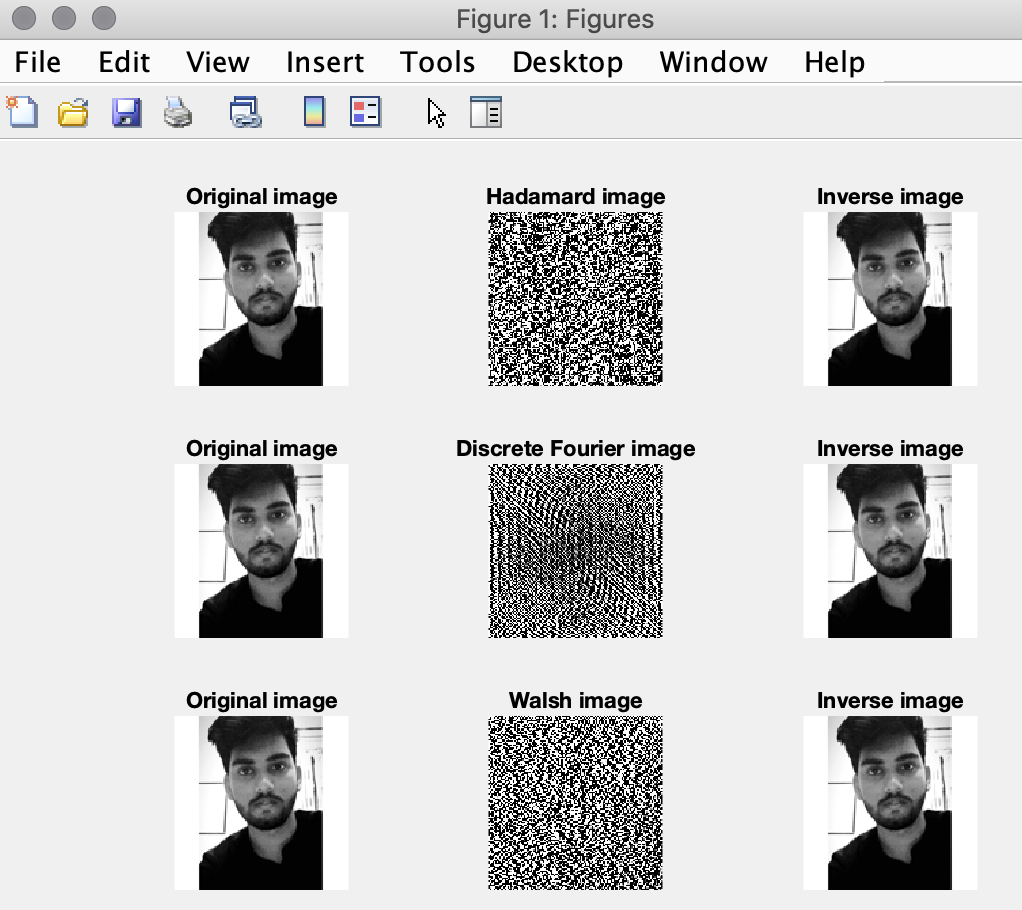
**B.2 Input and Output:**

**Input Images:**

****

**Output Images:**

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

****

**A picture containing book, people, computer, large

Description automatically generated**

**A close up of a piece of paper

Description automatically generated**

**B.3 Observations and learning:**

When working with images, sometimes operations in spatial domain do not have the desired output so the image is converted to transform domain, operations related to the domain are performed and converted back to spatial domain as the output of transform domain cannot be understood easily. Some operations in transform domain are Hadamard transform, Discrete Fourier transform and Walsh transform. Hadamard operation has a set 2x2 matrix using which we can find other transform matrices. Walsh transform matrix is Hadamard matrix arranged in increasing order of sign changes. Discrete Fourier has an inbuilt function that can be called to create the transformed image. All these transformed images will have to be converted back to spatial domain using inverse operation.

**B.4 Conclusion:**

I have understood the concept of transform domain and some operations done in that domain. I have understood the concept of and implemented Hadamard transform, Discrete Fourier transform, and Walsh transform as well as understood their effect on images.

**B.5 Question of Curiosity**

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

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

Answer:

* 1. Hadamard Transform is used in data encryption, as well as many signal processing and data compression algorithms. It can be used also to design error correcting block codes which can correct large numbers of errors in communications.

Discrete Fourier Transform can be used to calculate a signal's frequency spectrum and find a system's frequency response from the system's impulse response, and vice versa. It can also be used as an intermediate step in more elaborate signal processing techniques like in FFT convolution, which is an algorithm for convolving signals that is hundreds of times faster than conventional methods.

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

* 1. Discrete Cosine Transform is used for lossy data compression and works by separating the images into parts of differing frequencies. The DCT can be used to convert the signal (spatial information) into numeric data (frequency information) so that the image’s information exists in a quantitative form that can be manipulated for compression.

Steganography is the science of embedding information into the cover image viz., text, video, and image (payload) without causing statistically significant modification to the cover image. Image based steganography that combines Least Significant Bit (LSB), DCT, and compression techniques on raw images to enhance the security of the payload. Initially, the LSB algorithm is used to embed the payload bits into the cover image to derive the stego-image. The stego-image is transformed from spatial domain to the frequency domain using DCT. Finally, quantization and run length coding algorithms are used for compressing the stego-image to enhance its security.

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