|  |
| --- |
| Image and Video Processing Lab  **Name: -** Yash Rajput  **Registration Number :** 211060042 **Date: -** 09.02.2024 |
| Experiment 2  2D DFT |

**Aim: -** You are tasked with developing a program to compute 2D Discrete Fourier Transform (DFT) and comment on the results.

**Software Used: -** MATLAB

**Code:**

% IVP Lab – Experiment No: 2 – 2D DFT

% Yash Rajput – TY EC - 211060042

myArray = [];

disp('Entered output will be in this manner')

disp('[1 5 9 13]')

disp('[2 6 10 14]')

disp('[3 7 11 15]')

disp('[4 8 12 16]')

disp('Enter 16 numeric elements to append to the array:');

while length(myArray) < 16

% Prompt the user for input

userInput = input('Enter a numeric element: ', 's');

% Convert the input to a number (if possible)

numInput = str2double(userInput);

% Check if the input is a valid number

if ~isnan(numInput)

% Append the number to the array

myArray(end+1) = numInput;

disp(['Element ', num2str(length(myArray)), ' added to the array.']);

else

disp('Invalid input. Please enter a valid numeric number.');

end

end

matrix = reshape(myArray, 4, 4);

disp('Original Matrix:');

disp(matrix);

disp('Press 1 for Discrete Fourier Transform (DFT)')

disp('Press 2 for Discrete Cosine Transform (DCT)')

disp('Press 3 for Wavelet Transform')

choice = input('Enter your choice: ');

if choice == 1

% DFT

type = 'DFT of Matrix';

transformed\_matrix = fft2(matrix);

elseif choice == 2

% DCT

type = 'DCT of Matrix';

transformed\_matrix = dct2(matrix);

elseif choice == 3

% Wavelet Transform

type = 'Wavelet Transform of Matrix';

[c,s] = wavedec2(matrix, 2, 'db1');

c = transformed\_matrix;

else

disp('Invalid Choice')

end

% Display the original matrix and its DFT

disp('Original Matrix:');

disp(matrix);

disp(type);

disp(transformed\_matrix);

subplot(2,1,1)

imshow(matrix, []);

colormap(gray);

title('Original Matrix');

subplot(2,1,2)

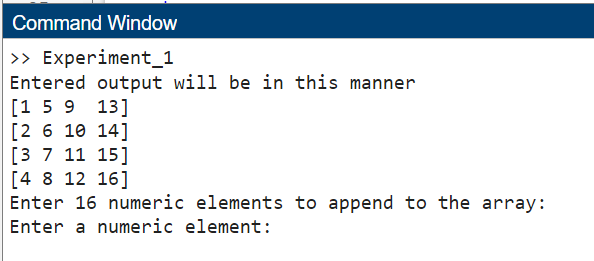
imshow(abs(transformed\_matrix), []);

colormap(gray);

title(type);

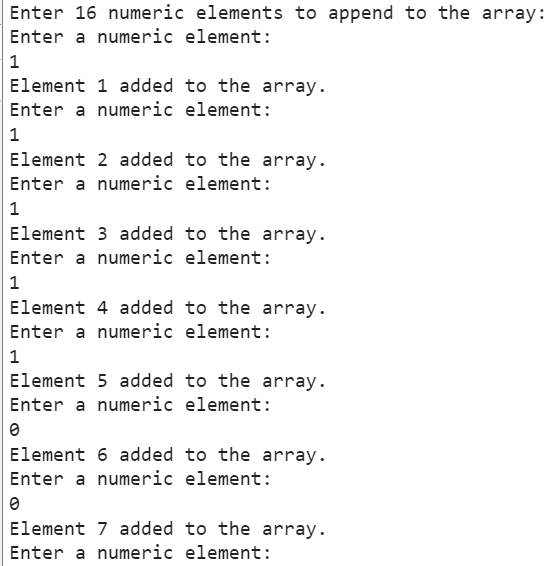
**User Interface :**

**Step 1:** The code begins with asking the user for the input matrix

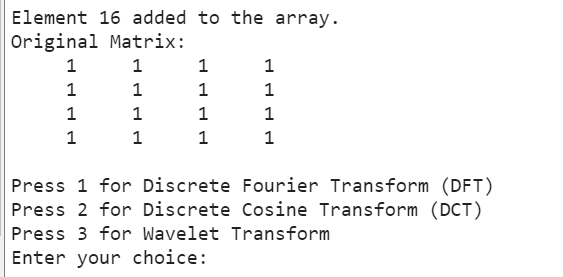
****

**Step 2:** Assuming there is not error in the input all the elements need to be entered one by one according to the sample matrix

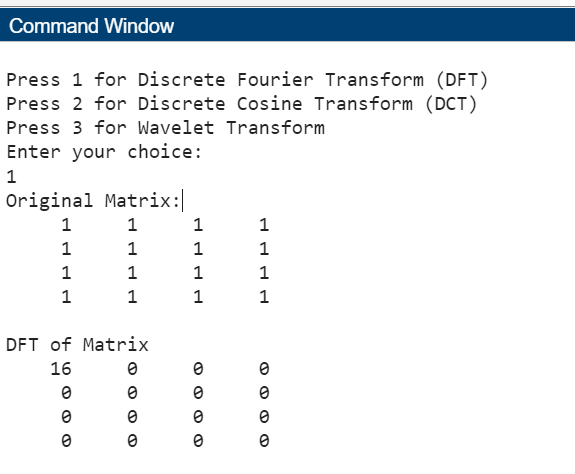
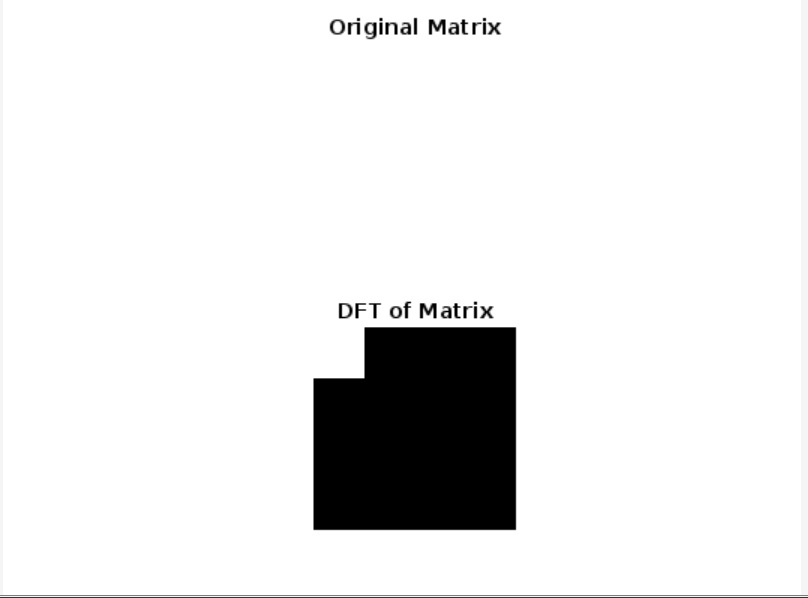
**Note:** All the errors related to input are explained in the Error Handling Section

****

**Step 3:** You will be shown your entered matrix and will be asked to choose the transformation you want to see



**Step 3:** After Entering your choice you can see the original matrix and the transformation in the Command Window and see its visualization in figure

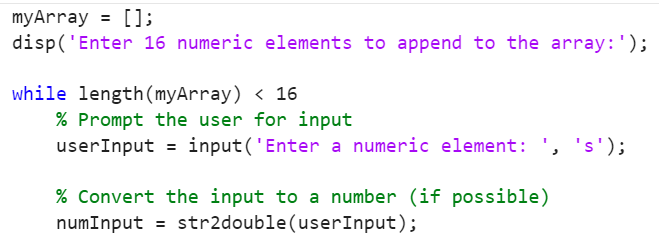
**Error Handling:**

**Case 1: The Entered Matrix is asymmetric**

# 

# The case of matrix being asymmetric cannot occur as we are collecting 16 elements in an array and then reshaping it into a 4x4 matrix

**Case 2: The Entered Matrix is not of 4x4 order**

****

# To deal with the input not having 16 elements, the code will simply not come out of the while loop until the user has entered 16 valid numbers

**Case 3: The Entered Matrix contains non numeric number**

# 

# Here the code will check each input before appending to the array, if the input is a valid number then it will append it otherwise it will skip it and the while loop will run until 16 numbers are completed

**Output Analysis**

# Test Case 1: Black Image

# 

# Visual Representation:

# 

# Conclusion: We can see that the original image and DFT of the image are same as the image does not contain any bright points and due to which the brightness of the image comes out to be 0

# Test Case 2: White Image

# 

# Visual Representation:

# 

# Conclusion: The original matrix cannot be seen as it is all white and has blend with the background. Over here the DFT shows the maximum brightness which is 16 as it’s the sum of all the elements and is at the point which denotes lowest frequency

# Test Case 3: Horizontal Strips

# 

# Visual Representation:

# 

# Conclusion: Over here we can see the brightness as 8 which is the sum of the elements in the original image, and as the original image has alternate rows the DFT shows the change in rows

# Test Case 4: Vertical Strips

# 

# Visual Representation:

# 

# Conclusion: Over here we can see the brightness as 8 which is the sum of the elements in the original image, and as the original image has alternate columns the DFT shows the change in columns

# Test Case 5: Chess Pattern

# 

# Visual Representation:

# 

# Conclusion: Over here we can see the bright points in DFT of the image is of value 8 we can see it being at the top left corner and another point in approximately in middle indicating that there’s frequent changes in the original image

# Test Case 6: Discrete Cosine Transformation

# 

# Visual Representation:

# 

# Conclusion: Over here we can see the bright points in DCT of the image is of value 4 as we analyze only the cosine part of the image

# Test Case 7: Wavelet Transformation

# 

# Visual Representation:

# 

# Conclusion : wavelet decomposition of the matrix X at level N using the wavelet wname . The output decomposition structure consists of the wavelet decomposition vector C and the bookkeeping matrix S , which contains the number of coefficients by level and orientation.