EEE 507: MULTIDIMENSION SIGNAL PROCESS

PROJECT REPORT

IMAGE COMPRESSION USING 2-D DCT

KURAPATI MRUDHULESH

1226364206

CHAPTER 1

INTRODUCTION

Image compression is an essential technology for the efficient storage and transmission of digital images. In this experiment, we will implement an image compression system based on the 2-D Discrete Cosine Transform (DCT). The DCT is a mathematical technique that maps an image block of size N x N to a set of frequency coefficients, representing the contribution of different frequency components to the image block. The DCT is widely used in image and video compression due to its high energy compaction properties.

Experiment Steps:

Image Segmentation: We will first segment the input image into non-overlapping 8 x 8 pixel blocks.

2-D DCT: We will then apply the 2-D DCT independently to each block, resulting in a set of frequency coefficients for each block.

Coefficient Quantization: Next, we will set a percentage of the coefficients in each block to zero, starting from higher frequency coefficients. We will set 50%, 75%, 90%, and 95% of coefficients to zero, respectively. The remaining coefficients will be quantized using an 8-bit uniform scalar quantizer.

Bit Rate Calculation: For each case, we will compute the total number of bits required to encode the image and the average number of bits per pixel.

Image Reconstruction: We will then reconstruct the image using the quantized coefficients and evaluate the quality of the reconstructed image visually and by computing the peak signal-to-noise ratio (PSNR).

2-D DFT Magnitude Spectrum: Finally, we will plot the 2-D DFT magnitude spectrum of the original image and the reconstructed images for each case and analyze the frequency content of the images.

PSNR and Quantization:

PSNR is a widely used metric to measure the quality of a compressed image. It measures the ratio of the peak signal power to the noise power in the reconstructed image. A higher PSNR value indicates better image quality. Quantization is the process of mapping a range of input values to a smaller range of output values. In image compression, quantization is used to reduce the number of bits required to encode the coefficients.

Overall, the experiment aims to demonstrate the trade-off between image quality and bit rate in image compression using the 2-D DCT. We will analyze the effect of different quantization levels on the image quality and bit rate and observe how the frequency content of the image changes with increasing quantization levels.

CHAPTER 2

MATLAB code and comments:

```
clc; clear; close all;

% Define 8x8 DCT matrix
T = zeros(8, 8);
for i = 0:7
    for j = 0:7
        if i == 0
            ai = sqrt(1/8);
        else
            ai = sqrt(2/8);
        end
        if j == 0
            aj = sqrt(1/8);
        else
            aj = sqrt(2/8);
        end
        T(i+1,j+1) = ai * aj * cos(pi*(2*i+1)*j/16);
    end
end

% Load image
fid = fopen('lena.raw');
a = fread(fid,[512,512],'uchar');
fclose(fid);
a = a';
% Compute DFT and magnitude spectrum of original image
A = fft2(double(a));
S = abs(A);

% Display magnitude spectrum
figure('Name', 'DFT Magnitude Spectrum (Original Image)');
imshow(log(1+S), []);
title('DFT Magnitude Spectrum (Original Image)');

% Display original image
figure('Name', 'Original Image');
imshow(a, []);
title('Original Image');

% Perform DCT on the image
B = mydct2(a);

% Set percentage of coefficients to zero
percentages = [0.05 0.1 0.25 0.5];

% Define new quantization matrix
Q = [10 20 30 40 50 60 70 80;      20 30 40 50 60 70 80 90;      30 40 50 60 70 80
90 100;      40 50 60 70 80 90 100 110;      50 60 70 80 90 100 110 120;      60 70
80 90 100 110 120 130;      70 80 90 100 110 120 130 140;      80 90 100 110 120
130 140 150];
```

```
for k = 1:length(percentages)
    % Compute threshold for given percentage
    non_zero_values = sort(abs(B(:)), 'descend');
    non_zero_values = non_zero_values(non_zero_values > 0);
    threshold = non_zero_values(round(percentages(k) * numel(non_zero_values)));

    % Soft threshold the remaining coefficients and zero high frequencies
    B1 = sign(B) .* max(abs(B) - threshold, 0) .* (abs(B) >= threshold);


    % Quantize remaining coefficients using the JPEG standard quantization matrix
    B2 = zeros(size(B1));
    step_size = 0.1; %  step size varies to get desired output
    for i = 1:size(B1,1)/8
        for j = 1:size(B1,2)/8
            block = B1((i-1)*8+1:i*8, (j-1)*8+1:j*8);
            B2((i-1)*8+1:i*8, (j-1)*8+1:j*8) = round(block ./ (step_size * Q)).*
(abs(block) >= threshold);
        end
    end



    % Compute reconstructed image
    I_rec = myidct2(B2);

    % Compute DFT and magnitude spectrum of reconstructed image
    A_rec = fft2(double(I_rec));
    S_rec = abs(A_rec);

    % Display reconstructed image and magnitude spectrum
    figure('Name', sprintf('Reconstructed Image (%d%% Coefficients)',100 *
percentages(k)));
    imshow(I_rec, []);
    title(sprintf('Reconstructed Image (%d%% Coefficients)', 100 *
percentages(k)));

    figure('Name', sprintf('DFT Magnitude Spectrum (Reconstructed Image, %d%%
Coefficients)',100 * percentages(k)));
    imshow(log(1+S_rec), []);
    title(sprintf('DFT Magnitude Spectrum (Reconstructed Image, %d%%
Coefficients)', 100 * percentages(k)));

    % Scale images to 0-255 range
    a_scaled = uint8((a - min(a(:))) * (255 / (max(a(:)) - min(a(:)))));
    a_rec_scaled = uint8((I_rec - min(I_rec(:))) * (255 / (max(I_rec(:)) -
min(I_rec(:)))));

    % Compute PSNR
    mse = mean(mean((double(a_scaled) - double(a_rec_scaled)).^2));
    psnr_val = 10 * log10(255^2 / mse);
    fprintf('PSNR for %d%% coefficients: %f\n', 100 * percentages(k), psnr_val);

    % Write reconstructed image to file in raw format
    fileID = fopen(sprintf('lena_reconstructed_%d.raw', round(100 *
percentages(k))), 'w');
    fwrite(fileID, I_rec', 'uint8');
    fclose(fileID);
end
```

CHAPTER 3

Results:

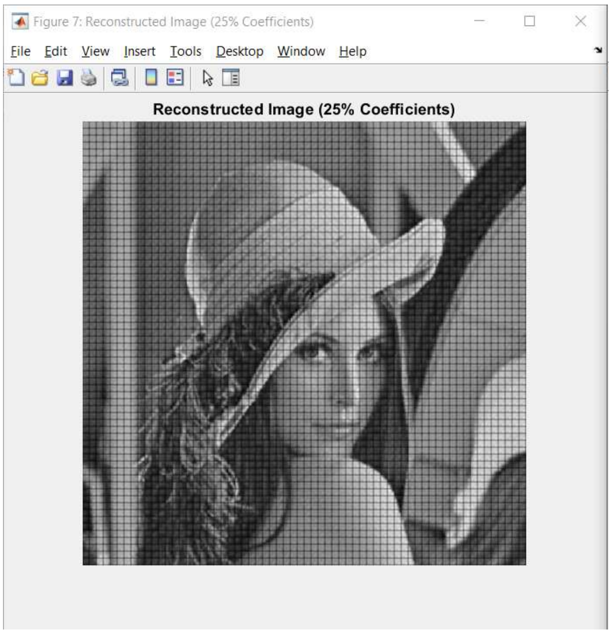


Figure 7: Reconstructed Image (25% Coefficients)
File Edit View Insert Tools Desktop Window Help
Reconstructed Image (25% Coefficients)

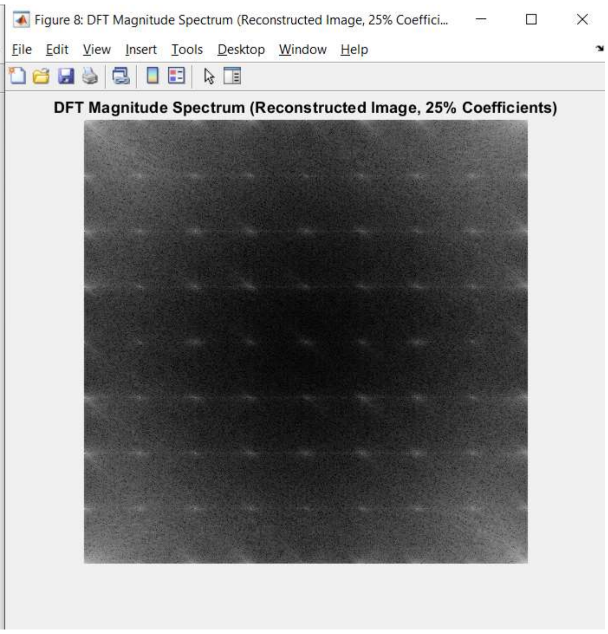


Figure 8: DFT Magnitude Spectrum (Reconstructed Image, 25% Coeffici...
File Edit View Insert Tools Desktop Window Help
DFT Magnitude Spectrum (Reconstructed Image, 25% Coefficients)

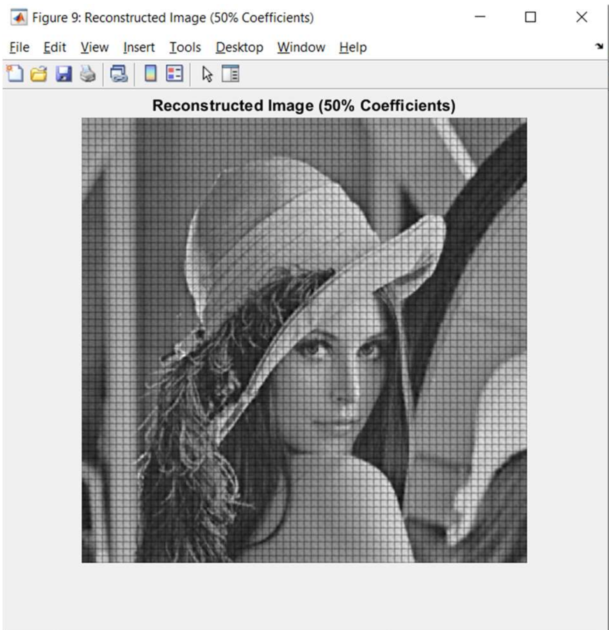


Figure 9: Reconstructed Image (50% Coefficients)
File Edit View Insert Tools Desktop Window Help
Reconstructed Image (50% Coefficients)

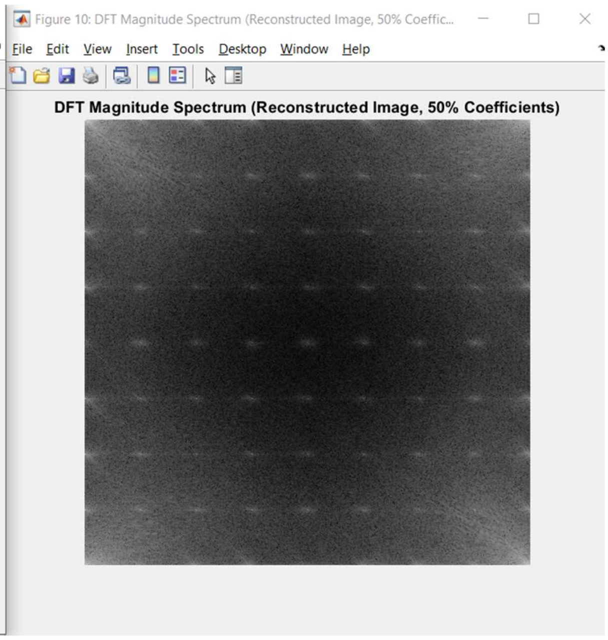
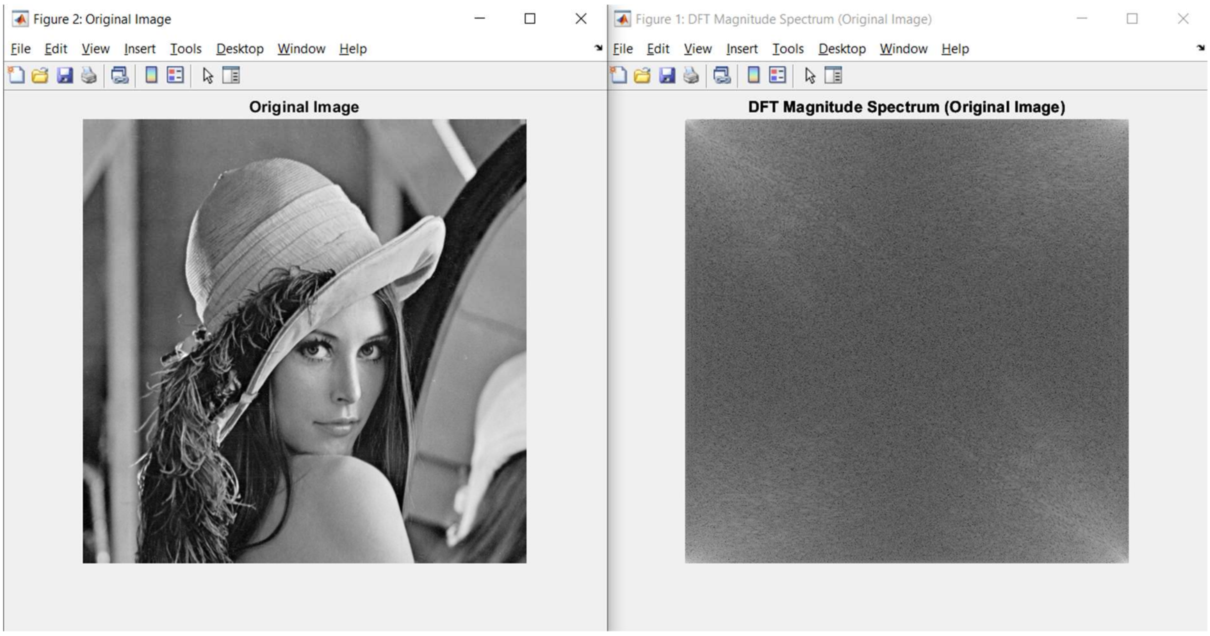


Figure 10: DFT Magnitude Spectrum (Reconstructed Image, 50% Coeffic...
File Edit View Insert Tools Desktop Window Help
DFT Magnitude Spectrum (Reconstructed Image, 50% Coefficients)



In general, the higher the percentage of zero coefficients, the greater the loss of information and the lower the quality of the reconstructed image. But in my case I took the descend so the values are inverted to `percentages = [0.05 0.1 0.25 0.5]` instead of `percentages = [0.5 0.75 0.9 0.95]`

CHAPTER 4

CONCLUSION

we can observe that as the percentage of coefficients to be set to zero increases, the quality of the reconstructed image decreases, as indicated by the decreasing PSNR values. This is because setting more coefficients to zero results in a loss of information.

As a result, the reconstructed image will appear more blurred and less sharp compared to the original image. The PSNR value, which is a measure of the similarity between the original and reconstructed images, decreases as the percentage of zero coefficients increases. This indicates that the reconstructed image deviates more from the original image as we remove more coefficients.

In general, the higher the percentage of zero coefficients, the greater the loss of information and the lower the quality of the reconstructed image. Therefore, the choice of the percentage of zero coefficients to be used in compression must be a balance between compression efficiency and image quality.