**CompE565, Spring 2022**

**JPEG based Image Compression**

**Prepared by**

*John Sadiq (825764388), Dustin Nguyen (824783655)*

E-mail: *JSadiq0352@SDSU.EDU, dnguyen2468@sdsu.edu*

Electrical and Computer Engineering

San Diego State University

**Table of Contents**

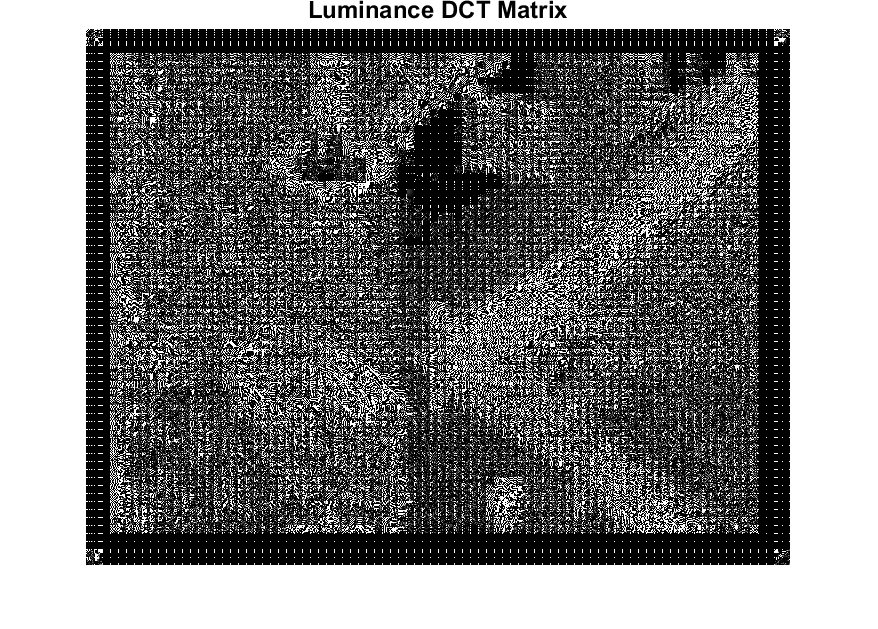
| Number | Title | Page number |
| --- | --- | --- |
| 1 | **List of figures** | 2-4 |
| 2 | **Introduction** | 5 |
| 3 | **Procedural Section** | 5 |
| 4 | **Results** | 5 |
| 5 | **Conclusion** | 7 |
| 6 | **References** | 7 |

**List of figures**

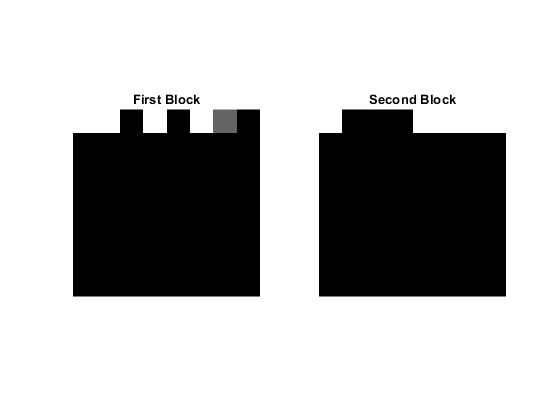
1. Original image



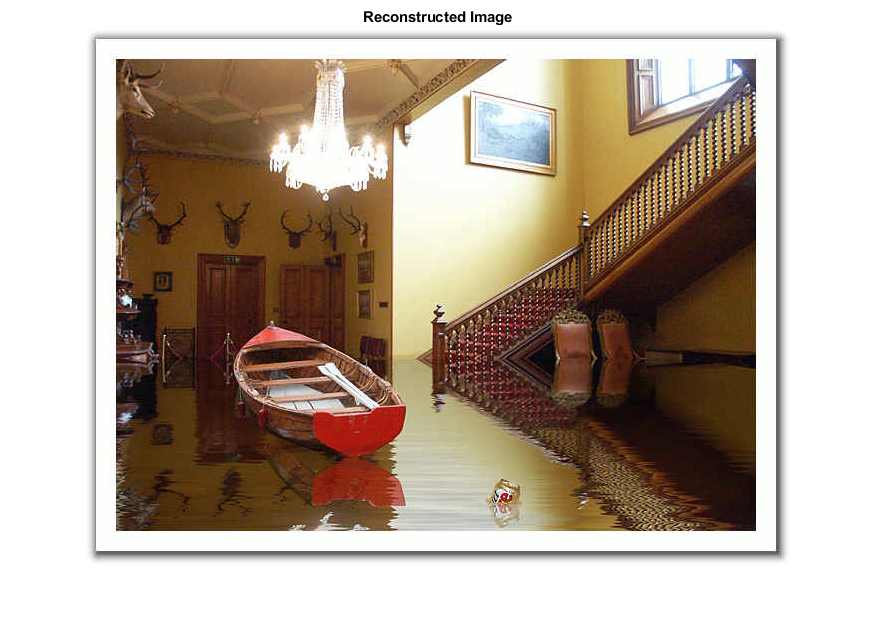
2) Luminance DCT



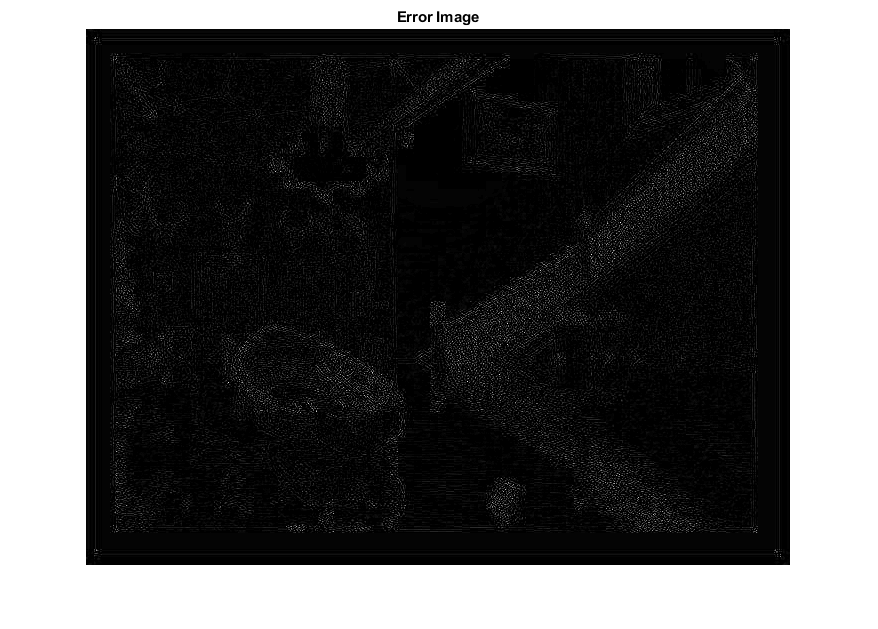
3) First 2 blocks in the 6th row (of blocks) from top for the luminance component



4) Reconstructed Image



5) Error image



**Introduction**

Multimedias have become a huge part of our lives in today’s society especially with the introduction of newer technologies such as smartphones, easier internet access, and better recording devices. However, the higher quality of multimedias (pictures and videos) do come with a cost, larger file size. Although the amount of digital memory has been improved a lot (smaller hardware with more memory space), the demand of memory space for high quality multimedias is still very high and it becomes a big problem. To solve this, the engineers have came up with a solution, file compression. In this HW, we will be using Matlab to compress an image file by subsampling the image and DCT/Quantization. First, we will write an Encoder to compress the image, then we will write a Decoder to reconstruct the compressed data to be a new image that will look exactly like the original image (to the human eye) but with smaller file size.

# **Procedural Section**

The approach to this assignment is very straight forward, an image can be compressed using subsampling and DCT/Quantization in Matlab. Then, a new reconstructed image can be produced using the compressed data from the processes above. The result should be a smaller file size image without any differences to the original image when observed with naked eyes.

The process is as followed:

* Read the image from a directory on local computer and show it using Matlab to make sure the image can be read.
* Convert the image from RGB to YCbCr and obtain each element separately (Y, Cb, Cr).
* Subsampling to 4:2:0.
* Write an Encoder - process image in 8x8 blocks, convert each block into frequency domain by using DCT2, get the quantized value for each block using Quantization matrices provided.
* Extract the first two blocks in the 6th row (of blocks) from top for the luminance component and display them (shown in Figure 3).
* Create a zigzag scanning function (“zigzag”) and use the function to generate AC DCT coefficient matrix and display along with DC DCT coefficient for both blocks extracted.
* Write a Decoder to restore the image from compressed data - use the built-in block processing functions to create required variables for the decoder, inverse quantization, inverse DCT, and upsample the image using Row/Column Replication.
* Calculate MSE and PSNR using the formula given in lectures.

# **Results**

By using the processes above, a new smaller size image was successfully reconstructed from compressed data and the difference in quality compared to the original can’t be seen by human eyes. The result is the same quality image with smaller required memory space.

Below are the DCT coefficients of the first two blocks in the 6th row from the top for the luminance component of the image.

firstBlock = 1.0e+03 \*

1.6880 0.1472 -0.0323 0.0142 -0.0060 0.0043 0.0004 -0.0010

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

secondBlock = 1.0e+03 \*

1.7120 -0.2099 -0.1457 -0.0611 0.0180 0.0669 0.0820 0.0525

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

First block DC DCT coefficient: 1688.00

First block AC DCT coefficient matrix:

105 0 0 0 0 0 0 0

13 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

-3 0 0 0 0 0 0 0

1 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

Second block DC DCT coefficient: 1712.00

Second block AC DCT coefficient matrix:

107 0 0 0 0 0 0 0

-19 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0

0 0 0 2 0 0 0 0

0 0 0 1 0 0 0 0

-15 0 0 0 0 0 0 0

-4 1 0 0 0 0 0 0

0 2 0 0 0 0 0 0

MSE =

11.9147

PSNR =

37.3700

# **Conclusion**

Image/Video compression algorithm, these days, is a must have when dealing multimedias because without a compression algorithm, a lot of memory space is required to store multimedia data (more money invested in memory) and streaming will not be as fast when the size of the file is large.

The compression method done in this assignment works as intended, however, it is not as effective as many other improved algorithms that we currently have.

# **References**

* HW1
* Matlab Commands PDF
* Zig-Zag Scan (MathWorks)  
  <https://www.mathworks.com/matlabcentral/fileexchange/27078-zig-zag-scan>