Final Project Report
9 December 2024

1 Introduction

This report presents an analysis of a Discrete Cosine Transform (DCT)-based image compression algorithm applied to three sample images: *rpi_86_field*, *rpi_empac*, and *rpi_office_for_research*. The objective is to evaluate the effectiveness of the compression technique by examining the file size reduction, computational overhead, and visual quality of the reconstructed images.

Central to this methodology are the mathematical operations—most notably the DCT, its inverse (IDCT), and the quantization step—that together enable significant data reduction with minimal perceptual loss. In this report, we clarify the role of these equations and how they inform both the compression ratio and the resulting image quality.

2 Methodology

The compression process involves the following steps:

1. **Color Space Conversion:** The input image is read in RGB format and converted to YCbCr. This allows separate handling of luminance (Y) and chrominance (Cb, Cr) and more efficient compression due to the human eye's lower sensitivity to color detail.

2. **Block Partitioning:** The luminance channel is divided into 8×8 blocks. This block-wise approach localizes computations and leverages the DCT to represent image data in terms of spatial frequencies.

3. **Discrete Cosine Transform (DCT):** Each 8×8 block undergoes a 2D DCT, transforming spatial pixel values into frequency-domain coefficients. Low-frequency coefficients generally capture most of the visually significant information, while high-frequency coefficients represent finer details and noise.

4. **Quantization:** After computing the DCT coefficients, quantization is applied to reduce the precision of less visually important (often higher-frequency) coefficients. By dividing each DCT coefficient by a corresponding quantization factor and rounding to the nearest integer, many high-frequency coefficients become zero, effectively discarding some detail and achieving data reduction.

5. **Inverse DCT and Reconstruction:** The quantized coefficients are dequantized, and the IDCT is applied to reconstruct each 8×8 block back into the spatial domain. Although not a perfect reconstruction, this step yields an approximation that is visually close to the original block.

6. **Saving the Compressed Image:** The compressed image is saved (e.g., as a JPEG), and its file size is recorded. Execution time and file size differences are measured for further analysis.

3 Equations and Metrics

Discrete Cosine Transform (DCT)

The 2D DCT of an 8×8 block is defined as:

$$X_{u,v} = \alpha(u)\alpha(v) \sum_{x=0}^{7} \sum_{y=0}^{7} x_{x,y} \cos\left(\frac{\pi(2x+1)u}{16}\right) \cos\left(\frac{\pi(2y+1)v}{16}\right)$$

where $x_{x,y}$ is the pixel intensity at position (x, y) in the spatial domain, and $X_{u,v}$ is the corresponding DCT coefficient at frequency coordinates (u, v). The scaling factors $\alpha(u)$ and $\alpha(v)$ are defined as:

$$\alpha(u) = \begin{cases} \frac{1}{\sqrt{8}}, & u = 0 \\ \frac{1}{2}, & u > 0 \end{cases}$$

This equation projects the image block onto a set of cosine basis functions. Lower values of u, v correspond to low spatial frequencies (smooth variations), while higher values correspond to high spatial frequencies (fine details, edges, and noise).

Inverse DCT (IDCT)

Once quantized coefficients are processed, the IDCT is applied. The IDCT essentially reverses the DCT operation by reassembling the spatial domain image block from the frequency-domain coefficients. Mathematically, it follows the same structure as the DCT but applies the inverse transform. By doing so, we retrieve an approximation of the original image block, albeit with some loss due to quantization.

Quantization

Quantization maps a range of coefficient values to a smaller set of representable levels. If $X_{u,v}$ is a DCT coefficient and $Q_{u,v}$ is the corresponding quantization factor from the quantization matrix, the quantized coefficient $X'_{u,v}$ is:

$$X'_{u,v} = \text{round}\left(\frac{X_{u,v}}{Q_{u,v}}\right)$$

This step is crucial for achieving compression. By increasing $Q_{u,v}$ for higher frequencies, these coefficients are more aggressively reduced, often becoming zero after rounding. This drastically reduces the amount of data.

Dequantization

Before IDCT, dequantization is performed:

$$\tilde{X}_{u,v} = X'_{u,v} \cdot Q_{u,v}$$

This recovers an approximation of the original DCT coefficient. The approximation may deviate from the true $X_{u,v}$ due to the rounding step in quantization.

Compression Ratio

A key metric for evaluating compression is the *compression ratio*, defined by:

$$\text{Compression Ratio} = \frac{\text{Original File Size}}{\text{Compressed File Size}}$$

A higher compression ratio indicates more significant data reduction. However, extremely high ratios can degrade image quality since more information is discarded.

4 Terminal Output and Data

Below is the terminal output obtained after running the compression program on the three images:

```
Processed rpi_86_field.png
Original Image Size: 1021115 bytes
Compressed Image Size: 360215 bytes
Time taken: 1.84 seconds
------------------------------
Processed rpi_empac.png
Original Image Size: 859470 bytes
Compressed Image Size: 269191 bytes
Time taken: 1.66 seconds
------------------------------
Processed rpi_office_for_research.png
Original Image Size: 174895 bytes
Compressed Image Size: 52910 bytes
Time taken: 0.32 seconds
------------------------------
```

From these results, we can approximate the compression ratios:

$$\text{rpi\_86\_field:} \frac{1021115}{360215} \approx 2.83, \quad \text{rpi\_empac:} \frac{859470}{269191} \approx 3.19, \quad \text{rpi\_office\_for\_research:} \frac{174895}{52910} \approx 3.30.$$

5 Results and Analysis

Table 1 summarizes the results:

Image	Original (bytes)	Compressed (bytes)	Time (s)	Ratio
rpi_86_field.png	1021115	360215	1.84	2.83
rpi_empac.png	859470	269191	1.66	3.19
rpi_office_for_research.png	174895	52910	0.32	3.30

Table 1: Summary of compression results.

The compressed images are significantly smaller, with roughly a 2.8x to 3.3x reduction in size. The time taken scales with image dimension and content complexity. The results show that applying the DCT followed by quantization and subsequent IDCT allows for a meaningful trade-off between storage savings and image fidelity.

6 Visual Comparison

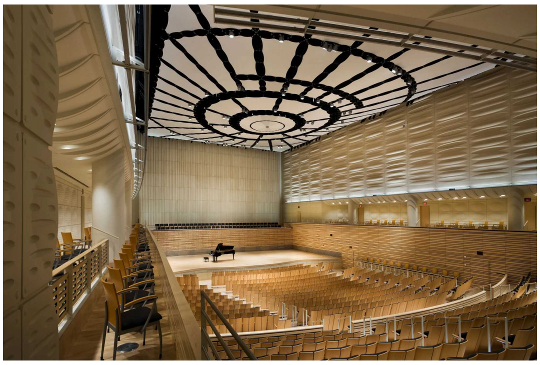
Below are side-by-side comparisons of the original and compressed images. Although the quantization step discards some high-frequency detail, the resulting images remain visually similar. The chosen quantization matrix and levels balance the removal of imperceptible details with the need to retain essential features.

(a) Original rpi_86_field

(b) Compressed rpi_86_field

Figure 1: Comparison of original and compressed images for rpi_86_field

(a) Original rpi_empac



(b) Compressed rpi_empac

Figure 2: Comparison of original and compressed images for rpi_empac

(a) Original rpi_office_for_research

(b) Compressed rpi_office_for_research

Figure 3: Comparison of original and compressed images for rpi_office_for_research

7 Conclusion

This project demonstrates that applying the DCT, quantization, and IDCT in a carefully designed pipeline can achieve substantial file size reductions. By expressing images in terms of spatial frequencies, quantizing away high-frequency components, and then reconstructing them, the algorithm provides compression ratios of approximately 2.8 to 3.3 while maintaining visually acceptable quality.

The equations governing DCT and quantization form the mathematical backbone of this approach. The DCT equation distributes energy into fewer critical low-frequency coefficients, while quantization equations systematically reduce the precision of less important data. The IDCT then serves as the bridge back to the spatial domain, resulting in a reasonable approximation of the original image. The compression ratio equation offers a straightforward measure of efficiency, reflecting how successfully the technique reduces data size without overly compromising image quality.

Overall, this DCT-based compression method effectively leverages mathematical transformations to provide a balanced compromise between storage, bandwidth, and perceptual quality.