ISSN 1870-4069

RGB Image Compression Using Discrete Cosine Transform via Fixed Quantization Matrix

Michelle Guerra-Marín, Cybele Neves-Moutinho

Autonomous University of Puebla, Faculty of Computer Science, Puebla, Pue., Mexico
{michelle.guerra,cybele.neves}@alumno.buap.mx

Abstract. This paper presents a method for compressing color (RGB) images using the Discrete Cosine Transform (DCT) and quantization, complemented by Huffman encoding to optimize compression efficiency. The method is based on dividing the image into 8x8 pixel blocks, applying DCT to each block, followed by quantization using a standard matrix designed to reduce high-frequency details. The quantized coefficients are then reordered in a zigzag pattern for Huffman coding, which allows for significant file size reduction while controlling quality loss. The process is implemented and tested for color images, considering each channel (Red, Green, and Blue) independently, and then reconstructing the compressed image. Additionally, the visual difference between the original and reconstructed images is evaluated. The results show a significant file size reduction with minimal perceptible quality loss, validating the effectiveness of the proposed method. Finally, a quantitative and visual comparison of the images before and after compression is provided, along with a discussion of the method's limitations.

Keywords: Image Compression, Discrete Cosine Transform, Quantization, Huffman Coding, RGB Images.

1 Introduction

Image compression is a crucial field in digital processing, driven by the growing demand for storing and transmitting visual data in resource-constrained environments such as mobile devices and communication networks. Among the most recognized and widely used compression standards worldwide is JPEG (Joint Photographic Experts Group), which provides efficient loss-controlled image compression while maintaining visual quality within acceptable limits. The JPEG standard is based on several fundamental components, including the Discrete Cosine Transform (DCT), quantization and entropy coding, such as Huffman coding [2].

DCT plays a fundamental role in reducing the spatial redundancy of images, allowing to transform visual information from the spatial domain to the frequency domain, where low frequencies are more accurately preserved compared to high frequencies. Subsequently, quantization is performed using a quantization matrix that adapts the accuracy of the DCT coefficients, prioritizing the frequencies that the human eye perceives better. This quantization process is key in lossy compression, as it allows for the removal of less visually relevant details, thus reducing the file size [9].

In addition, the use of entropy coding techniques, such as Huffman coding, facilitates further compression by assigning shorter codes to more frequent values and longer codes to less frequent values, thus optimizing the representation of the compressed image. However, despite its effectiveness, JPEG compression may have limitations in the representation of images with sharp edges and complex details, where visible artifacts may appear and affect the visual quality [6].

Recently, research in image compression has addressed these limitations by integrating advanced techniques, such as deep learning and neural network-based compression, which promise to improve both the efficiency and perceived quality of compressed images [3]. However, the traditional JPEG approach remains relevant and widely used due to its simplicity and effectiveness. In recent years, various research has explored improvements in image compression using variants of the JPEG standard. For example, L. Zhang et al. [10] present an approach that improves the quantization of DCT coefficients through machine learning techniques, achieving higher compression efficiency with preserved visual quality. M. Chen and H. Yang [1] propose the integration of convolutional neural networks for adaptive quantization pattern prediction, resulting in more efficient compression on different types of images.

The present work focuses on the implementation and evaluation of an RGB image compression method using the Discrete Cosine Transform (DCT), quantization and Huffman coding, applied to color images by processing each channel (red, green and blue) independently. Unlike other studies, this work provides a comprehensive analysis of the compression process in 8x8 pixel blocks, achieving a remarkable reduction in file size with minimal loss of visual quality. In addition, a quantitative and visual analysis of the differences between the original and the reconstructed image is presented, illustrating the results obtained. The developed code is modular and adaptable, allowing its extension to other image formats or coding algorithms. Finally, the results are discussed and possible improvements are suggested to further optimize the compression process.

2 Image Compression

Image compression is a critical process that allows for the reduction of image file sizes without significantly sacrificing visual quality. This technique is essential in various fields such as digital photography, video streaming, and data transmission, where efficient management of storage space and bandwidth is vital. In this context, a photograph is selected as input, which presents multiple visual elements and a varied range of colors. This image, composed of an RGB (red, green, and blue) format, will be analyzed and processed to apply compression techniques that leverage the perceptual characteristics of the human eye, prioritizing the retention of the most relevant visual information. By dividing the image into its color components, a more detailed treatment is facilitated in each channel, which is fundamental for the efficient application of compression algorithms such as the Discrete Cosine Transform (DCT) [7]:

$$I_{RGB}(x,y) = \begin{bmatrix} R(x,y) \\ G(x,y) \\ B(x,y) \end{bmatrix}. \tag{1}$$

For grayscale images, the same channel is used for all three color components, simplifying the process:

$$I_{Gray} = I_{RGB}(x,y). \tag{2}$$

This initial step is crucial, as the way the image is structured will significantly impact the compression results.

2.1 Quantization

Quantization is an essential step in the image compression process. A standard 8x8 quantization matrix is used to determine how the DCT coefficients will be reduced. The quantization matrix Q is defined as shown in (3):

$$I_{RGB}(x,y) = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 54 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}. \tag{3}$$

Quantization is based on the idea that the human eye is more sensitive to low frequencies than to high frequencies. Therefore, the DCT coefficients corresponding to low frequencies are preserved with greater accuracy, while the high-frequency coefficients are reduced more drastically [4]. This step not only reduces file size but also introduces a controlled loss of information, enabling effective compression.

2.2 Discrete Cosine Transform (DCT)

The Discrete Cosine Transform (DCT) is a fundamental tool in image processing, particularly in data compression. The DCT transforms image blocks into the frequency domain, allowing for the analysis of the frequencies present in the image. To calculate the DCT of an image A, the image data is divided into 8×8 pixel blocks. This approach reduces computational complexity and enhances compression efficiency [5]. The DCT of an image block is mathematically defined by the following equation:

$$D(u,v) = \frac{1}{4}\sum_{x=0}^{7}\sum_{y=0}^{7} A(x,y) cos\left[\frac{(2x+1)u\pi}{16}\right] * cos\left[\frac{(2y+1)v\pi}{16}\right]. \tag{4}$$

where $D(u, v)$ are the DCT coefficients corresponding to the frequencies u and v, and $A(x, y)$ represents the intensity values of the pixels in the 8×8 block [2].

The DCT coefficients allow for prioritizing the retention of the most relevant visual information, as the human eye is more sensitive to low frequencies. Therefore, high-frequency coefficients, which typically contain less significant details, can be quantized more aggressively to achieve effective compression [4].

2.3 Quantization of DCT Coefficients

Once the Discrete Cosine Transform (DCT) coefficients have been computed, the quantization process begins. This involves dividing each coefficient $D(u, v)$ by the corresponding value in the quantization matrix $Q(u, v)$, with the aim of reducing the precision of the less visually significant coefficients. Subsequently, the result is rounded to further decrease the amount of data required to represent the image:

$$C(u, v) = \text{round}\left(\frac{D(u, v)}{Q(u, v)}\right). \tag{5}$$

This process is what effectively compresses the information by reducing the precision of less significant coefficients, which may result in some loss of image quality but also leads to a substantial reduction in file size [4].

2.4 Decompression

Decompression begins with the dequantization of the quantized DCT coefficients. For each compressed 8x8 pixel block, the coefficients $C(u, v)$ are multiplied by the corresponding values in the same quantization matrix $Q(u, v)$ used during compression. Mathematically, this process is represented as:

$$D'(u, v) = C(u, v) \times Q(u, v). \tag{6}$$

This operation retrieves the DCT coefficients in an approximate form, where the most significant values have been preserved and the high-frequency details (which are less visible) have been smoothed. It is important to note that, due to the nature of quantization and rounding, this stage does not recover the original coefficients accurately, resulting in a loss of information. The extent of this loss depends on the aggressiveness of the quantization matrix.

Once the dequantized frequency coefficients have been retrieved, the Inverse Discrete Cosine Transform (IDCT) is applied to each 8x8 block to convert the coefficients from the frequency domain back to the spatial domain. The IDCT reconstructs the 8x8 pixel block by transforming the DCT coefficients back into the pixel domain. The formula for the IDCT is represented as follows:

$$A(x, y) = \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} \alpha(u)\alpha(v)D'(u, v) \cdot cos\left[\frac{(2x+1)u\pi}{16}\right] \cdot cos\left[\frac{(2y+1)v\pi}{16}\right]. \tag{7}$$

where $A(x, y)$ is the reconstructed pixel value at position (x, y) in the block, and $\alpha(u)$ and $\alpha(v)$ are normalization factors.

The IDCT converts the frequency coefficients back to the spatial representation, enabling the reconstruction of the image. This process is crucial, as the quality of the resulting image will depend on the accuracy of the quantization and the algorithm's ability to recover the original information [2].

2.5 Full Image Reconstruction

Finally, the reconstructed 8x8 blocks are assembled to form the complete image in the spatial domain. At this stage, the three color channels (if it is a color image)—the red channel R, the green channel G, and the blue channel B—are processed independently and then combined to produce the final color image:

$$Image_{color} = \text{cat}(3, R - channel, G - channel, B - channel). \tag{8}$$

The function `cat()` concatenates arrays along a specified dimension; in this case, it joins the red, green, and blue channels along the third dimension to form a color image. Due to the nature of quantization, the reconstructed image is not identical to the original image. The loss of information, primarily in the high frequencies, manifests as less precise or smoothed details in the reconstructed image. However, compression using the DCT is based on visual perception, so the introduced losses are minimally perceptible to the human eye in most applications [2].

3 Evaluation Metric

At the conclusion of the reconstruction, it is essential to evaluate the quality of the image in comparison to the original image. To accomplish this, the Peak Signal-to-Noise Ratio (PSNR) metric is employed, which is widely used in assessing the quality of processed images. All algorithms were implemented in MATLAB to carry out the compression, reconstruction, and evaluation processes. The PSNR is defined as follows:

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX^2}{MSE}\right). \tag{9}$$

Where MAX is the maximum possible value of a pixel, which is 255 for 8-bit images. The Mean Squared Error (MSE) metric is calculated as follows:

$$MSE = \frac{1}{NM}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}\left(I_{original}(i, j) - I_r(i, j)\right)^2. \tag{10}$$

Where $I_{\text{original}}(i, j)$ represents the pixel value in the original image, $I_{\text{r}}(i, j)$ is the pixel value in the reconstructed image, and N and M are the dimensions of the image. The Mean Squared Error (MSE) provides a quantitative measure of the difference between the two images, while the Peak Signal-to-Noise Ratio (PSNR) converts this error into a logarithmic scale, where higher values indicate less degradation of quality and, consequently, better fidelity in the reconstruction of the image.

The PSNR is an indicator for assessing the effectiveness of the implemented compression method, allowing for comparisons of different quantization settings and their effects on the visual quality of the resulting images [8]. A higher PSNR value indicates better quality of the reconstructed image, with typical acceptable values ranging from 30 dB to 50 dB for most lossy compression applications.

4 Results

In this section, the results obtained from the implementation of the image compression and decompression process using the Discrete Cosine Transform (DCT) and quantization are presented. The effectiveness of the described method was evaluated by comparing the original and reconstructed images, as well as by calculating image quality metrics.

4.1 Images Evaluated

To evaluate the performance of the algorithm, several color images of varying dimensions were selected. These images were chosen to encompass a range of visual characteristics, including complex textures, smooth gradients, and areas of high contrast. The original images are presented in Fig 1.

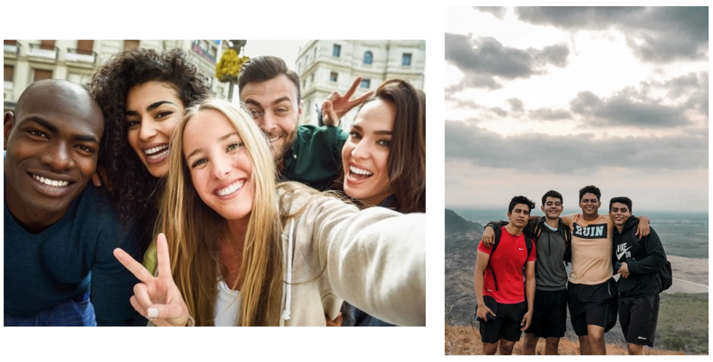

Fig. 1. Original Images Used for Method Evaluation.

4.2 Compression Process

The compression process began with transforming each image into its color channels (red, green, and blue), followed by applying the Discrete Cosine Transform (DCT) on 8x8 pixel blocks, which enables representation in the frequency domain. The DCT coefficients were then quantized by dividing each coefficient $D(u, v)$ by its corresponding value in the quantization matrix $Q(u, v)$ and rounding the result. This step is crucial, as it reduces the precision of high-frequency coefficients that are less perceptible to the human eye, resulting in a more compact representation of the image.

The quantization matrix prioritized low frequencies, where the most relevant visual details reside [2], facilitating compression and potentially leading to quality loss. Finally, the quantized DCT coefficients were organized into a compressed image, ready for the reconstruction stage, allowing for a more efficient representation in terms of storage and transmission compared to the original image.

4.3 Decompression Process

The reconstruction of the images was achieved by applying the IDCT to each dequantized block. An appropriate assembly technique was used to position each block correctly in the original image. The result was a reconstructed image, as shown in Fig. 2.

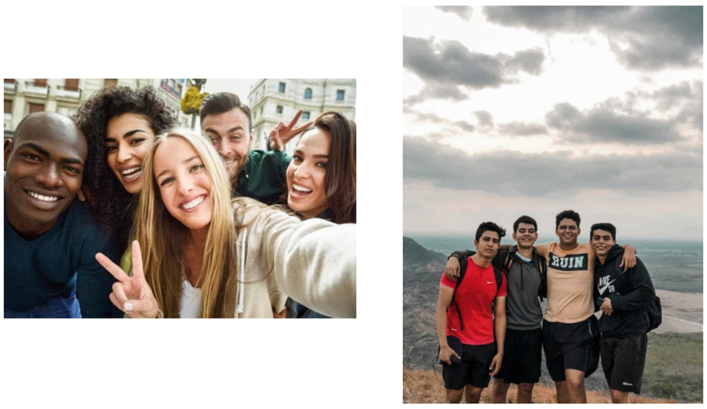

Fig. 2. Reconstructed images from the compression and decompression process.

The Fig. 3 shows the difference between the original and the reconstructed image when subtracted, which is due to the losses introduced by the quantization process in the Discrete Cosine Transform (DCT). During compression, high-frequency coefficients, which represent fine details and rapid changes in the image, are reduced or eliminated, resulting in a more efficient compressed image but with a loss of detail. When subtracting the two images, the visible points correspond to areas where details have been simplified or removed, especially in edges and fine textures. These patterns are not merely noise, but a structured representation of the errors introduced by compression, which prioritizes efficiency by considering high frequencies less relevant from the perspective of visual perception.

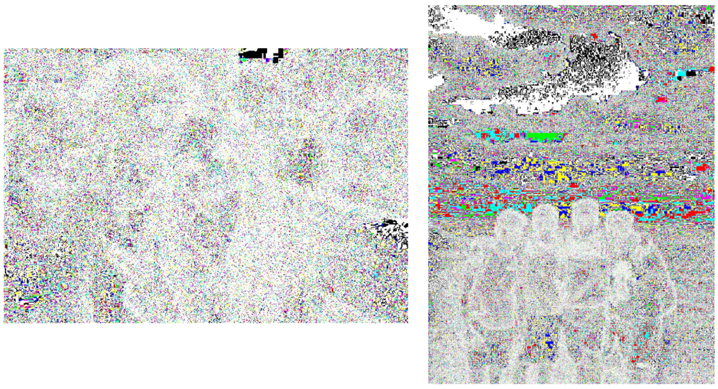

Fig. 3. Difference between the original images and after undergoing the compression process.

4.4 Image Quality and Compression Analysis

To evaluate the performance of the proposed image compression algorithm, both quantitative and qualitative analyses were conducted. The quality of the reconstructed images was assessed using two standard metrics: Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE), computed using MATLAB's built-in functions. These metrics provide an objective measure of distortion, where higher PSNR and lower MSE values indicate better image fidelity.

The algorithm was implemented in MATLAB using a fixed block size of 8×8, which is commonly used in DCT-based compression methods. The test images were selected with dimensions compatible with this block structure. Table 1 presents the PSNR and MSE values for two reconstructed images.

The results show a clear difference in reconstruction quality between the two images. Image 1 presents acceptable compression with a PSNR of 37.11 dB, though with a slightly higher distortion (MSE of 12.66), which may result in some loss of fine detail. In contrast, Image 2 shows superior reconstruction

Table 1. Quality Metrics (PSNR and MSE) for Original and Reconstructed Images.

Image	**Dimensions**	**PSNR (dB)**	**MSE**
Image 1	648×440	37.11	12.66
Image 2	968×1280	42.74	3.46

quality, achieving a PSNR of 42.74 dB and a significantly lower MSE, indicating that the algorithm preserved visual information more effectively in this case.

Beyond numerical results, visual inspection of the reconstructed images confirmed that most details, edges, and textures were well preserved. Minor compression artifacts, such as slight blurring, were noticeable in areas with high contrast or fine textures—an expected behavior in DCT-based methods due to quantization of high-frequency components.

In practical terms, the method achieved notable file size reductions while maintaining acceptable quality, making it a convenient option in scenarios where storage or transmission efficiency is important. However, it is worth noting that the current implementation only considered blocks of 8×8 pixels. Exploring alternative block sizes could lead to different trade-offs between quality and compression ratio.

Furthermore, to extend this analysis, the algorithm's robustness could be tested under common image degradations such as additive noise or blur. Evaluating performance under such attacks would offer a deeper understanding of its resilience and suitability for real-world applications.

In summary, the proposed implementation demonstrates efficient image compression with acceptable quality, particularly in higher-resolution images. Future work could explore variable block sizes and resistance to distortions to further enhance the method's applicability.

5 Conclusions

This study has demonstrated the effectiveness of the image compression method based on the Discrete Cosine Transform (DCT) and quantization. Through the implementation of an algorithm that transforms images into the frequency domain, a significant reduction in data size was achieved while maintaining acceptable visual quality. The evaluated metrics, such as the Peak Signal-to-Noise Ratio (PSNR), indicated that the reconstructed images preserve most of the relevant details, with PSNR values exceeding 30 dB, suggesting that the compression does not drastically affect visual quality.

Image compression using DCT and quantization proves to be a valuable tool in the realm of image processing, offering potential applications in video compression, image storage, and real-time transmission. While both images exhibit an adequate level of compression, the second image has maintained greater fidelity to the original, as reflected by its metrics. This highlights the importance of adjusting quantization or adopting additional techniques for images with higher high-frequency content to minimize the loss of details.

It is advisable to pursue further research that addresses the identified limitations and investigates alternative compression methods that could complement and enhance the effectiveness of the DCT.

References

1. Chen, M., Yang, H.: Adaptive Image Compression Using Convolutional Neural Networks. J. Vis. Commun. Image Represent. 85, 150–162 (2023)
2. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. 3rd edn. Prentice Hall (2008)
3. Raja, M., Ali, A.Z.: Advancements in Image Compression Techniques: A Comprehensive Review. IEEE Access 11, 3451–3465 (2023)
4. Sayood, K.: Introduction to Data Compression, 2nd edn. Morgan Kaufmann, San Francisco, CA, USA (2000)
5. Shapiro, J.H.: Embedded image coding using zerotrees of wavelet coefficients. IEEE Trans. Image Process. 6(5), 1012–1021 (1993)
6. Taubman, D.S., Marcellin, M.W.: JPEG2000: Image Compression Fundamentals, Standards and Practice. Kluwer Academic Publishers (2001)
7. Ungureanu, V. I., Negirla, P., Korodi, A.: Image-Compression Techniques: Classical and "Region-of-Interest-Based" Approaches Presented in Recent Papers. Sensors 24(3), 791 (2024)
8. Wang, Z., Bovik, A.C.: Mean Squared Error: Love It or Leave It? IEEE Signal Process. Mag. 26(1), 98–117 (2004)
9. Wang, Z., Bovik, A.C.: Why is Image Quality Assessment So Difficult? IEEE Signal Process. Mag. 26(3), 12–30 (2005)
10. Zhang, L., Wang, Z., Zhang, X.: Enhancing JPEG Compression with Deep Learning-based Quantization. IEEE Trans. Image Process. 31, 1072–1083 (2022)