Indian Institute of Technology, Ropar

CS517: Digital Image Processing and Analysis
Karanraj Mehta
2021CSB1100

JPEG Compression

April 19, 2023

Overview

This project aims to implement the JPEG compression algorithm for grayscale and color images. The algorithm mainly involves block-based division, DCT transformation, quantization and zigzag scanning. The project involves varying the number of coefficient parameters, sub-image/block size, and quantization matrices to analyze the impact on image compression and quality.

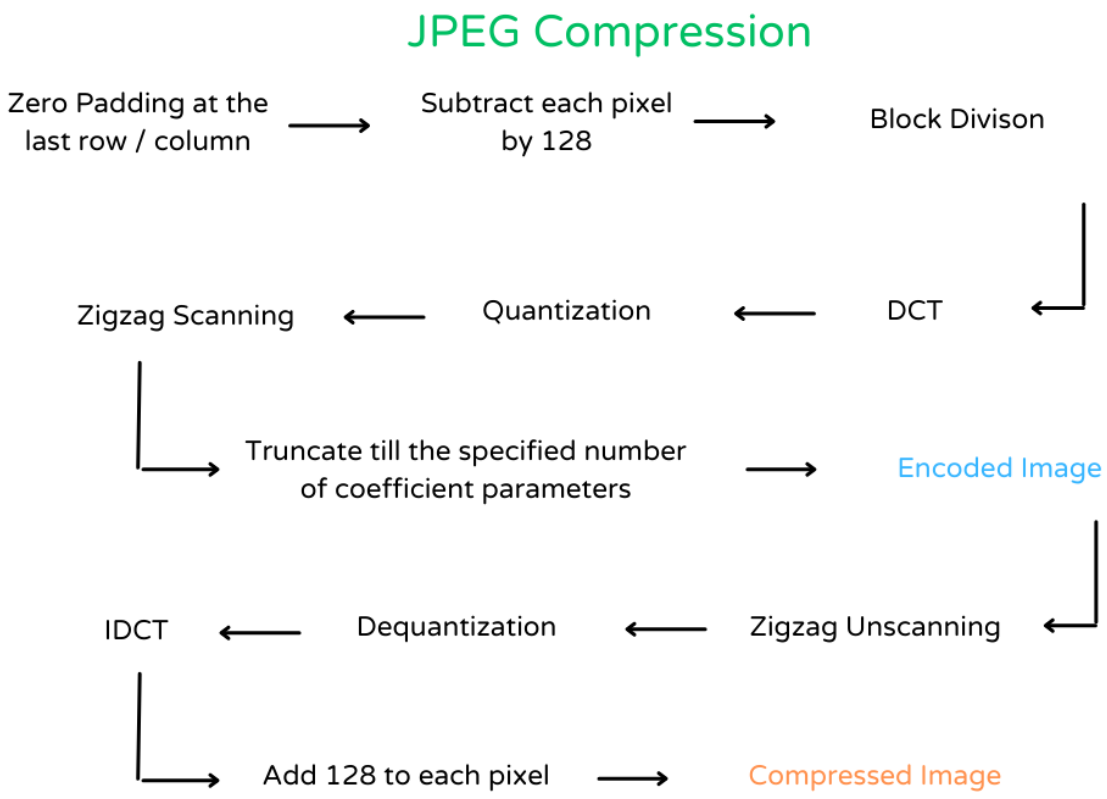
The project involves calculating the PSNR and Compression ratios and plotting a graph between the PSNR and Compression ratios for various values of the number of coefficient parameters for the given datasets.

Goals

1. **To implement a JPEG compression algorithm using OpenCV Python.**
2. **To analyze the effect of parameters such as the number of coefficient parameters used, sub-image/block size, normalization/quantization matrix, etc., on the compression ratio and PSNR.**
3. **To implement a JPEG encoder and decoder for grayscale and colored images**
4. **To plot a graph between PSNR and compression ratio for different values of the number of coefficient parameters for the provided datasets.**

Specifications

The program consists of utility functions and analysis functions. The JPEG encoding process for grayscale images involves padding the image with zeros in the last row/column to make it divisible by the user-defined block size. Next, we subtract each pixel in the padded image by 128, and we divide the resulting image into blocks of size block_size x block_size, rounding up the width/height to the nearest multiple of block_size if necessary.



Each block is then transformed using cv2.dct and divided by the standard quantization matrix. The resulting values are rounded, and each block is traversed in a zigzag order to obtain a list of 1D arrays corresponding to each block. This list is truncated to the number of coefficient parameters specified by the user. This is the JPEG-encoded array.

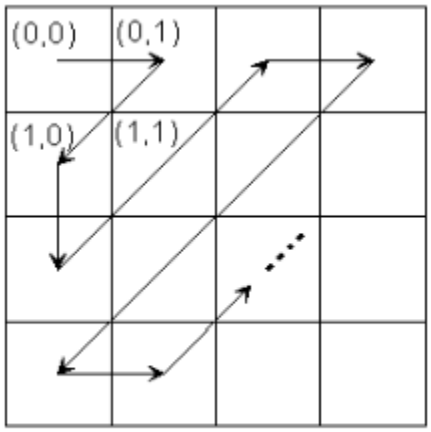


Fig. 1. Zigzag order

For grayscale decoding, the process is the reverse of the encoding process. Each 1D array corresponding to a block is used to obtain a 2D block of size block_size x block_size using the zigzag_unscan utility function. The quantized blocks are obtained by multiplying each block with the quantization matrix. We then take the IDCT of each block, add 128 to it, and round it. Placing these blocks back in their respective positions yields the compressed image.

The user-defined block_size must be an even number to satisfy the requirement of cv2.dct. To use the standard quantization matrix for different block sizes, we resize it using cv2.resize.

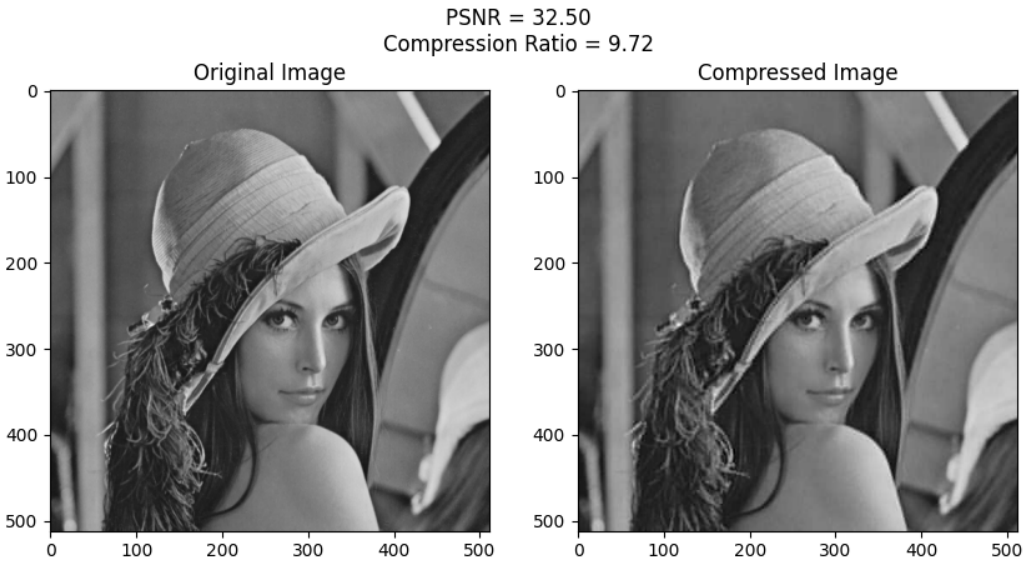
In case of color images, the image is divided into its color channels, and the grayscale JPEG encoding is applied to each channel to obtain the corresponding JPEG encoded arrays. For color image decoding, we decode the JPEG encoded array for each channel and then merge them to reconstruct the colored image.

This is just a high-level description of the program. There are many finer details which have been explained within the code through comments.

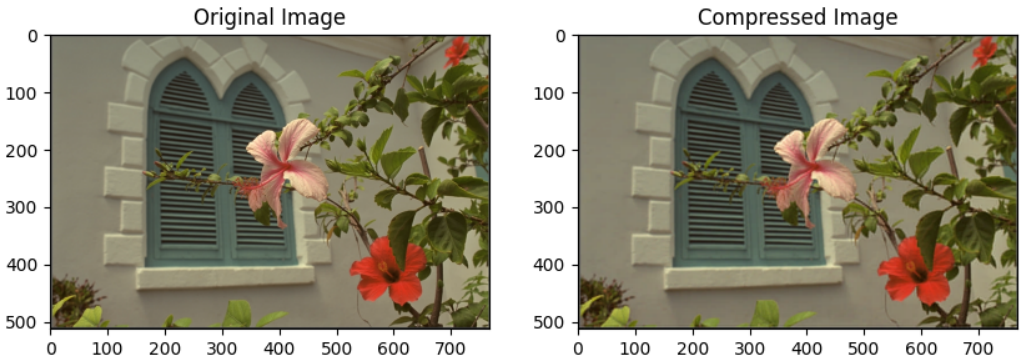
Analysis and Results

The program includes two main functions, jpeg_encoder and jpeg_decoder, as described in the problem statement. Additionally, three analysis functions have been created. The first, analyze_image, returns a tuple of the original image, compressed image, PSNR, and compression ratio. By adjusting parameters such as block_size and num_coefficents, the effect on PSNR and compression ratio can be analyzed.

The plot_images function takes an image and uses matplotlib to plot it alongside the compressed image, displaying the PSNR and compression ratio. Two example images are shown: a grayscale image with block_size=8 and num_coefficents=10 and a colored image with the same parameters.



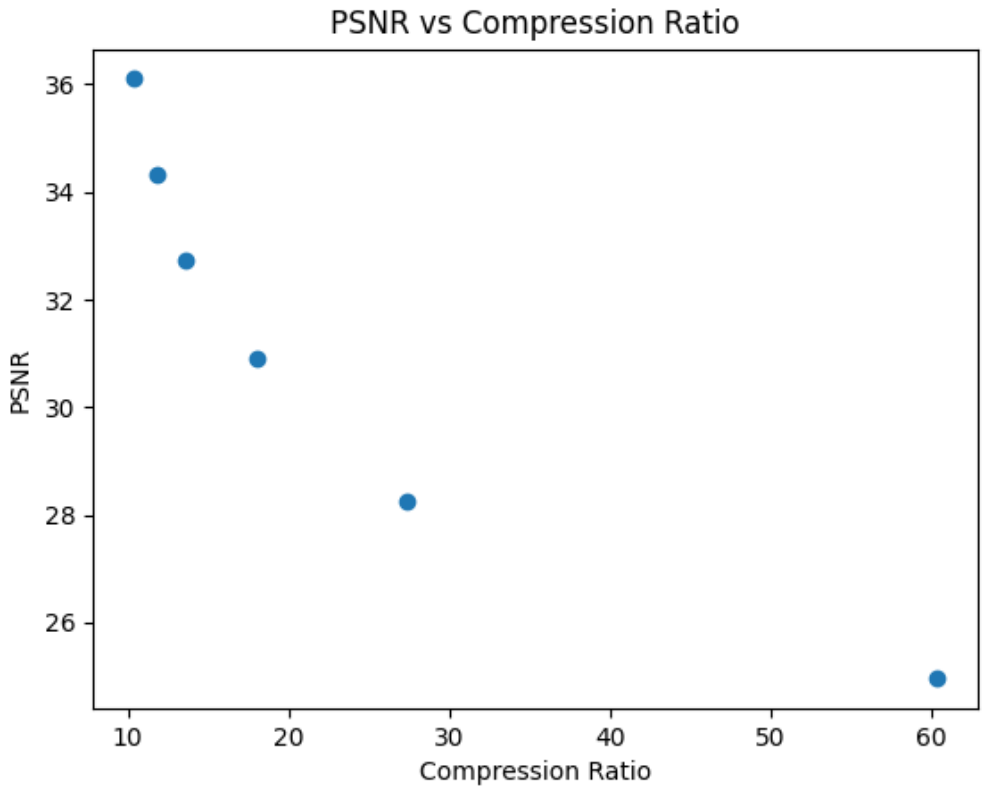
PSNR = 30.76
Compression Ratio = 9.48

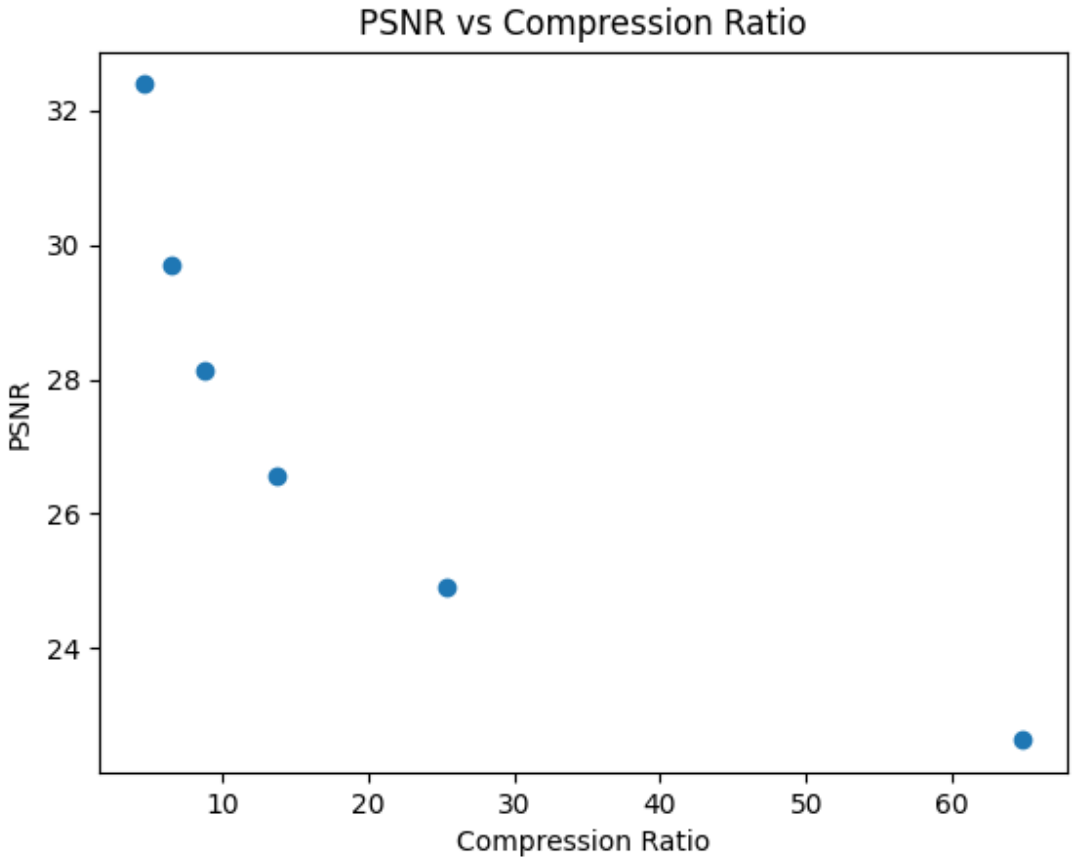


Colored image with block_size = 128 and num_coefficient = 10

Lastly, plot_graph has been created to plot a graph between the average PSNR and compression ratio for all images in a given image folder, with options for color and grayscale images. This graph plots values for number of coefficient parameters in 1, 3, 6, 10, 15, and 28. Graphs for both color and grayscale datasets are shown.

The graph illustrates the relationship between compression ratio and average PSNR for a grayscale image data set, with varying numbers of coefficients.





The graph illustrates the relationship between compression ratio and average PSNR for a color image data set, with varying numbers of coefficients.

Usage

To run this program, ensure you have Python version 3.10 or higher installed on your system. Additionally, you will need to install the following libraries: numpy, opencv, and matplotlib.

To test the code, go to the bottom of the file and replace the image path with the path to your desired image for the plot_images function. To run the plot_graph function and plot a graph between PSNR and compression ratio, replace the image folder path with the path to your sample images folder. Finally, run the program by executing the command "python A4_JPEG_2021CSB1100_Karanraj.py" in your terminal.