**JPEG COMPRESSION : A DEEP DIVE INTO DCT AND RUNLENGTH ENCODING**

Prepared for

Dr. Satvik Vats

Graphic Era Hill University

Prepared by

Gaurav Pandey(PV-21180566)

**15 Jan 2023**

TABLE OF CONTENTS

1. Introduction:
   * Image Processing and Compression Overview……………………………1
   * Popular Techniques: RLE, DCT, JPEG……………………………………1
2. Literature Review:
   * Digital Images and Image Processing……………………………………..2
   * Image Compression: Lossy vs. Lossless…………………………………..2
   * Popular Compression Techniques (JPEG, MP3, ZIP)……………………..3
   * JPEG Compression Algorithm……………………………………………..3
3. Methodology:
   * Color Space Conversion……………………………………………………4
   * Chrominance Downsampling………………………………………………5
   * Discrete Cosine Transform (DCT) & Quantization………………………..6
   * Run Length Encoding (RLE)…………………………………………….7-8
   * Decompression Process…………………………………………………….9
4. Pseudocode:
   * Steps to image compression………………………………………………10
5. Result:
   * Outcome of the Algorithm……………………………………………… 10
6. References ………………………………………………………………………..11
7. About us ………………………………………………………………………….12

INTRODUCTION:

Imagine giving human-like vision and an understanding of images to computers. Image processing and computer vision can help with it. Computer vision aids in the interpretation of visual data by machines, while image processing modifies and analyses images to improve them. While edge detection techniques separate things from their surrounds, noise reduction algorithms brighten photos that are a little too fuzzy. Image processing is used by facial recognition systems to recognize faces in crowded regions, while medical image analysis technologies provide the most accurate diagnosis of diseases and cancers.

However, visual data is frequently expensive. One high-resolution photo might contain gigabytes of data, which presents processing, transmission, and storage difficulties. This is when picture compression's magic kicks in. By using digital smart, methods like Run-Length Encoding (RLE) and Discrete Cosine Transform (DCT) can reduce the size of an image without sacrificing important information.

Effective data storage techniques are vital in a world where data is flowing in by the bucket. They aid in managing the overwhelming amount of information, maintaining scalability, affordability, and quickness. Simple storage techniques additionally guarantee that your data remains dependable and does not use excessive space. Additionally, everything runs more smoothly when it comes to things like images because of creative techniques for compressing them, including image compression.

We will explore deeper into the interesting field of image compression through this report, looking at its theoretical foundations, real-world uses, and continuous development. We will look at how the most popular image format, JPEG, uses RLE and DCT to accomplish its amazing achievements and investigate state-of-the-art compression methods that exceed the limits of efficiency and image quality.

LITERATURE REVIEW:

A digital image is an image composed of picture elements, also known as pixels, a pixel is the smallest controllable element of a picture represented on the screen, each with finite, discrete quantities of numeric representation for its intensity or any level that is an output from its 2-D functions fed as an input by its spatial coordinates denoted by x and y on the x-axis and y-axis respectively. The 2-D function can be written as f(x, y) where x and y are spatial coordinates. The amplitude of the f is called intensity or gray level at the point (x, y).

Image processing or Digital Image processing is the processing of image which are digital in nature. It is the use of a digital computer to process digital images through an algorithm. covers low, mid and high-level processes. In low Level input and output are images. Mid-level outputs are attributes extracted from input images. In high level an ensemble of recognition of individual objects is used. Images which are digital in nature have various advantages it gives us the improvement of pictorial information as it is clearly visual and information can be extracted very easily. For digital images the transmission is very fast and it gives us efficient storage.

An image is a representation of something or someone. For e.g.: any drawing, painting, photograph etc. Images are a very powerful tool in communication. Basically, there are three types of images- binary images, Gray-scale images and RGB colour images. Binary images (Fig1) contain only two values 0 and 1 and are usually black and white. Gray-scale images (Fig2) are where the pixel intensity ranges from 0 to 255, 0 represents the black colour and 255 represents the white colour. All the in-between values are different shades of gray. RGB colour images (Fig3), where “RGB” represents the Red, Green and Blue channels of an image. It contains 65,536 different colours which are a combination of RGB values ranging from 0 to 255 for each red, green and blue colours.

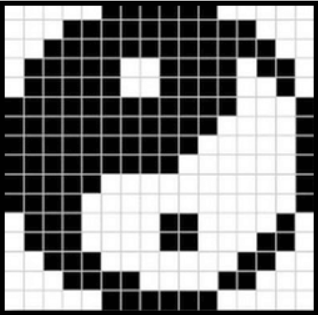
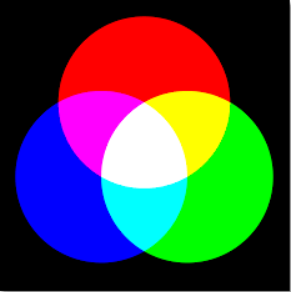
  

Fig : Binary Image Fig : Gray-scale image Fig : RGB image

A typical image is represented as a matrix values of this matrix corresponds to pixel intensity values. A larger number means brighter pixel and smaller number means darker pixel. Colour images have different channels for each colour components, such as red, green and blue although this is the most common way to represent an image, it's not how they are typically stored on a disk. Let’s say we have a 24 megapixel colour picture which means we have 12 million value to store for each colour channel leading to a total of 36 million values. If we assume that these values are stored as 8-bit integers we should end up with a 36 megabyte file.

Image compression becomes necessary at this point. The primary cause of image compression is image processing. Image compression is the technique of enlarging an image while maintaining its original quality. It makes an image smaller by removing characteristics that are redundant or impossible for the human eye to see. The main goal of this process is to eliminate as much data as possible while yet safeguarding the majority of the information. There are two types of methods used to compress a file: lossy and lossless. Lossy compression algorithm reduces the size of file by discarding the less important information in a file, which can significantly reduce the file size but also affect the file quality. The data discarded during lossy compression cannot be recovered. On the other hand, Lossless compression algorithm reduces the file size without losing any information, meaning we can reconstruct the original data from the compressed file.

Depending on what you require for a specific work, you can decide between lossless and lossy compression. Selecting the best technique is crucial because it requires achieving a balance between reducing file size and preserving crucial data. Each method has advantages and disadvantages. These days, there are sophisticated compression techniques that excel at this. They come up with clever solutions to meet the unique requirements of various sectors. Some of the most popular lossy compression techniques are JPEG (Joint Photographic Experts Group), MP3 (MPEG Audio Layer III), and video (e.g., H.264, H265), whereas lossless techniques include ZIP (and other archive formats), FLAC (Free Lossless Audio Codec), PNG (Portable Network Graphic), and GIF (Graphics Interchange Format).

77.8% of internet photographs are in the JPEG format, which is the most common image format for digital photography. This format uses lossy compression, translating spatial information into the frequency domain by applying the discrete cosine transform (DCT). Video formats like h.264, which are frequently used on websites like YouTube, can also be compressed using the same principles as JPEG. These technologies, which are essential in many fields, reduce the amount of storage that servers must handle and result in significant cost reductions. Even while JPEG isn't as complicated as other algorithms used in a variety of industries, learning it is nevertheless worthwhile.

The JPEG algorithm is a commonly used approach for photographing photos among other compression methods. It is a lossy compression method that considerably reduces the file size without sacrificing image quality. The JPEG technique can result in a picture that is ten times smaller than it was originally. It functions by eliminating information that is difficult for human eyes to see while keeping the details that human eyes are capable of identifying.

JPEG isn't ideal for all kinds of images. You may notice certain odd elements in a picture that has been converted to a JPEG format if it contains extremely clear details, writing, or odd patterns. Herein lies the role of researchers, who are constantly seeking to enhance or adjust JPEG.

METHODOLOGY

Broadly stating the process of compressing an image can be categorized into five steps which starts with Color space conversion where we convert the R,G,B values of each pixel to Y, Cb & Cr respectively here Y denotes luminance whereas Cb and Cr denotes chrominance the formula used for this conversion is stated later in this section. Color space conversion is followed by chrominance downsampling where we operate on the Cb and Cr frame individually, we divide the frame into blocks of 2\*2 and then calculate average value for each block, remove the repetitive information and shrink the block such that each average value of a four pixel block takes up a single pixel. Performing DCT(Discrete Cosine Transform) and Quantization on the matrix are the next steps followed by the last step Run length encoding .

Lets look at each of the steps mentioned above :-

**Colour space conversion:**

The original image consist of pixels and every pixel has a red, green and blue component each value from 0 to 255 and combination of these three value RGB results for a single pixel. The process of space conversion takes these three values and calculate three new values luminance, blue chrominance and red chrominance abbreviated as Y,Cb & Cr respectively this process is reversible and no data is removed during the conversion.

There value will always be in the range 0 to 255 YCbCr values can be calculated as

Y = 0.299 R + 0.587 G + 0.114 B

Cb = - 0.1687 R - 0.3313 G + 0.5 B + 128

Cr = 0.5 R - 0.4187 G - 0.0813 B + 128

This process is reversible and no data is lost. During decompression using these three linear equations, the value of respective RGB can be calculated.

**Chrominance down sampling:**

Chrominance downsampling operates on the Cb and Cr component of the image and removes repetitive information by converting the image into blocks of size 2\*2 and then calculating average value for each block, It shrink the block such that each average value of a four pixel block takes up a single pixel. By doing the above step the Cb and Cr frame size are reduced by one fourth and hence considerable size is reduced. When reassembling the picture the red and blue prominence images are rescaled to match the size of luminance component.

The fact behind the significance of Chrominance sampling is our eyes are bad at detecting color or chrominance and comparably good at perceiving Luminance because of the larger no of luminance perceiving photoreceptor cells(Rods) in our eyes which are around 20 times the Chrominance perceiving cells(cones) , Chrominance downsampling utilises the fact and down sample the chrominance attributes.

In the Fig4 & Fig5 we can see an example of the fact stated above :

Figure : Chrominance Figure :Luminance

**Discrete Cosine Transform:**

DCT, or Discrete Cosine Transform, is a technique that represents a finite sequence of data points as a sum of cosine functions oscillating at various frequencies. DCT work’s on the fact that our eyes are bad at perceiving high frequency elements within images. The discrete cosine transform and quantization step go through each section of the image and find areas that having high frequency of alternating chrominance or luminance. The elements which our eyes are not able to see are then removed. DCT or discrete Cosine Transform, is as technique

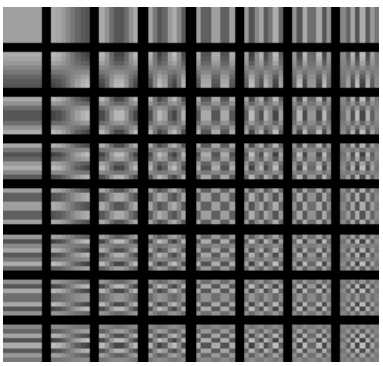
[](https://www.mathworks.com/help/images/discrete-cosine-transform.html)

Figure : 64 base images built from cosine functions

Let’s dive into it’s implementation, We perform the process on each of three components(Y,Cb,Cr) separately. The first step is to divide the entire image into 8\*8 sections called blocks each with 64 pixels with values 0-255 after that we subtract 128 from each pixel value and now the range of pixels is -128 to 127. In the next step we use 64 base images and express each block from our image as a combination of these 64 base images by plotting the frequency of each base image as per used. Thus the 64 pixel block each containing a value is formed into 64 values or constants that represents how much of each base image is used.

Let’s take a 8\*8 block (Figure 7)

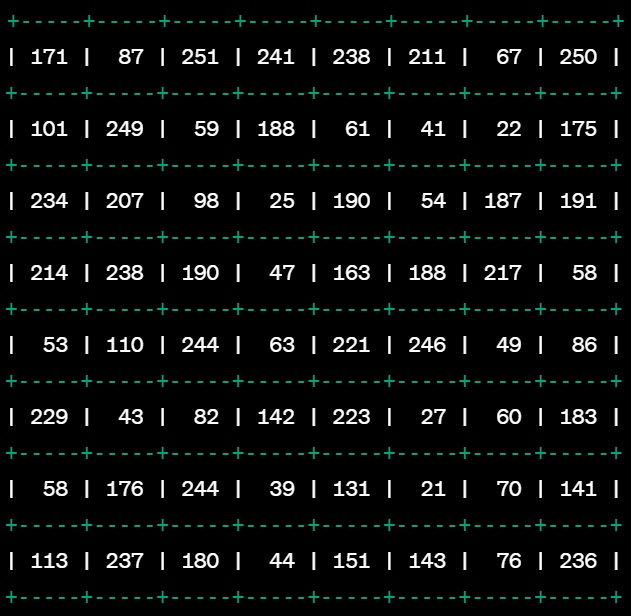


Figure : subimage

After subtracting 128(Figure 8)

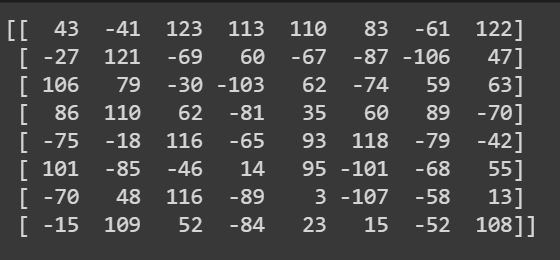


Figure 8: subimage after subtracting 128

After applying DCT the matrix obtained is:(Figure 9)

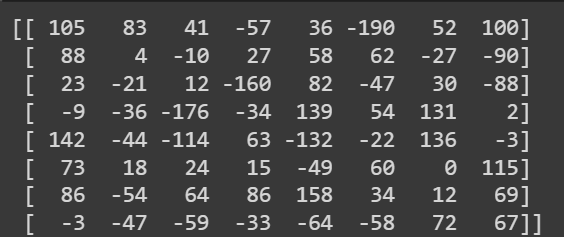


Figure 9: subimage after applying DCT

In DCT no data is lost, Data is lost in the following step Quantization. Let’s delve into Quantization.

**Quantization:**

We will now quantize the coefficient table we obtained using DCT. In quantization we have a table(Figure 10) of constants corresponding to utilization of each base image. Nothing in discrete cosine transform actually compresses or shrinks the image but quantization does.

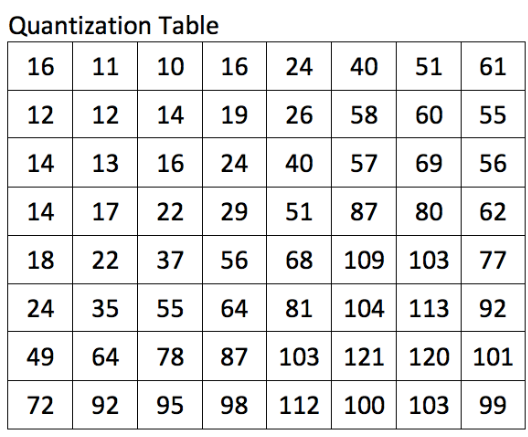
[](https://arjunsreedharan.org/post/146070390717/jpeg-101-how-does-jpeg-work)

Figure : The DCT Quantization Matrix Q

The next step is to divide each value in table of constants by corresponding values in the quantization table and round each value to the closest integer in the quantization table where the high frequency data is located in the right bottom corner where our eyes aren't great at perceiving and smaller number on the top left corner where distinct patterns are located. In this step we are throwing away the data that our eyes don’t perceive that’s why we are not able to tell the difference.

Here is the result of our matrix after performing quantization

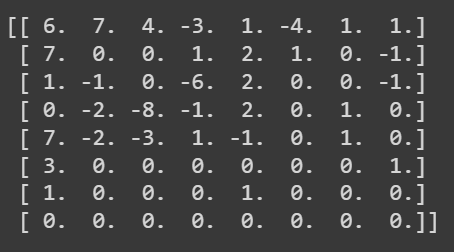


Figure 11: Matrix after performing Quantization

**Run length encoding**:

After performing quantization we have got the matrix which consist of repeating values mostly zeros so we will find a better way to store the subimage or block than before. We will store the data in an zig zag fashion, data of this patter can be easily compressed using Run length encoding algorithm. Here we instead of listing all the zeros we just store how many zeros there are. After this Huffman Encoding is also used to compress it even further but we have not used it in our program.

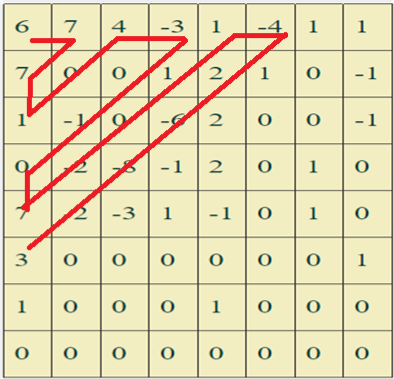


Figure : This data can be easily compressed using RLE

**Decompression:**

In the decompression process of a grayscale JPEG image, the compressed data, which includes the zigzag-scanned coefficients and may involve Run-Length Encoding, is first read and decoded. The zigzag ordering is then reversed to reconstruct the coefficients. Following this, the inverse quantization step is applied using the original quantization matrix to restore the frequency coefficients to their higher precision. Subsequently, the Inverse Discrete Cosine Transform (IDCT) is employed to revert the transformed image back to its spatial domain.

The decompressed grayscale image is then obtained, aiming to closely resemble the original. The absence of Huffman coding, as stated, simplifies the decoding process while retaining the efficiency achieved through quantization and the DCT. This tailored decompression procedure ensures optimal reconstruction of the grayscale image from

After all the decompressed image is saved as ‘NewImage.jpeg’ and we can easily see the artifacts present in the image that are due to lost data , further we will see the pseudocode of the whole process and of the way image compression was implemented .

PSEUDOCODE

1. Read the input grayscale image using OpenCV library(cv2.read()).
2. Define the block size and Quantization matrix(Standard Q matrix).
3. Create 8\*8 block divisions of the image.
4. Apply Discrete Cosine Transform on each block(used the cv2.dct() function.
5. Divide the resulting matrix with the Quantization matrix.
6. Perform zigzag scanning to reorder the coefficients.
7. Flatten the zigzag ordered coefficients and apply Run Length Encoding.
8. Write the compressed data to a file(“output.txt” in my case).
9. For decompression read the “output.txt” .
10. Retrieve the run-length encoded coefficients and dimensions.
11. Reconstruct the zigzag-ordered coefficients.
12. Apply inverse zigzag to obtain 8x8 blocks.

**RESULT:**

The image that was given to the program as input was “harry.jpg”

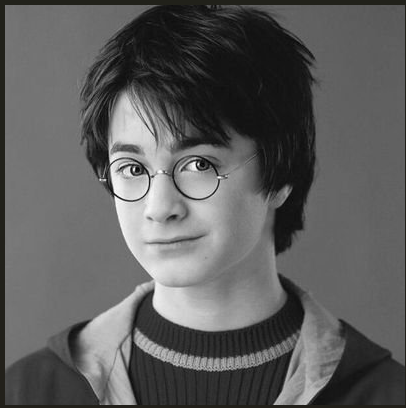
 

Figure : Harry.jpg(22kb)(Input) Figure : NewImage.jpg(16kb)(Output)

The image was compressed and the compressed data was stored in output.txt , after reading output.txt the image was decompressed and the resulting image “newharry.jpg” resulted which was of 16kb.

**REFERENCES :**

Websites(blogs) :

* <https://arjunsreedharan.org/post/146070390717/jpeg-101-how-does-jpeg-work>
* <https://www.sciencedirect.com/topics/computer-science/quantization-matrix>
* <https://unsplash.com/s/photos/binary>
* <https://www.cloudflare.com/learning/performance/glossary/what-is-image-compression/>
* <https://www.tablesgenerator.com/>

YouTube:

* [How are images compressed (Branch Education)](https://www.youtube.com/watch?v=Kv1Hiv3ox8I)
* [Lossy Image Compression : DCT(NPTEL)](https://www.youtube.com/watch?v=sckLJpjH5p8&t=1612s)
* [Introduction to Quantization(NPTEL)](https://www.youtube.com/watch?v=N1HnIERFgWk&t=12s)

GitHub:

* <https://github.com/amzhang1/simple-JPEG-compression>
* <https://github.dev/getsanjeev/compression-DCT>

**ABOUT US:**

Name : Gaurav Pandey

LinkedIn : <www.linkedin.com/in/gaurav-pandey-9a5b37229>

Contact : +91 6397939645