Visvesvaraya Technological University, Jnana Sangama, Belagavi – 590 018

KLS Vishwanathrao Deshpande Institute of Technology, Haliyal – 581 329

(Accredited by NAAC with "A" Grade)

Project Report

entitled

PERFORMANCE ANALYSIS AND FUNCTIONAL VERIFICATION OF DCT BASED LOSSY IMAGE COMPRESSION TECHNIQUE

Submitted in partial fulfillment of the requirement for the award of degree of

Bachelor of Engineering

in

Department of Electronics and Communication Engineering

Submitted by

Dhanya .V. Kulkarni	2VD21EC040
Pallavi .M. Betsur	2VD21EC061
Pooja .M. Betsur	2VD21EC064
Ramyashree .V.Nayak	2VD21EC071

VIII – Semester / IV – Year

Under the guidance of

Dr. Mahendra M. Dixit

Professor and Head of the Department

Academic Year 2024– 25 (Scheme 2021)

Visvesvaraya Technological University,
Jnana Sangama, Belagavi – 590 018

KLS Vishwanathrao Deshpande Institute of Technology,
Haliyal – 581 329

(Accredited by NAAC with "A" Grade)

Department of Electronics and Communication Engineering

Project Report

entitled

PERFORMANCE ANALYSIS AND FUNCTIONAL VERIFICATION OF DCT BASED LOSSY IMAGE COMPRESSION TECHNIQUE

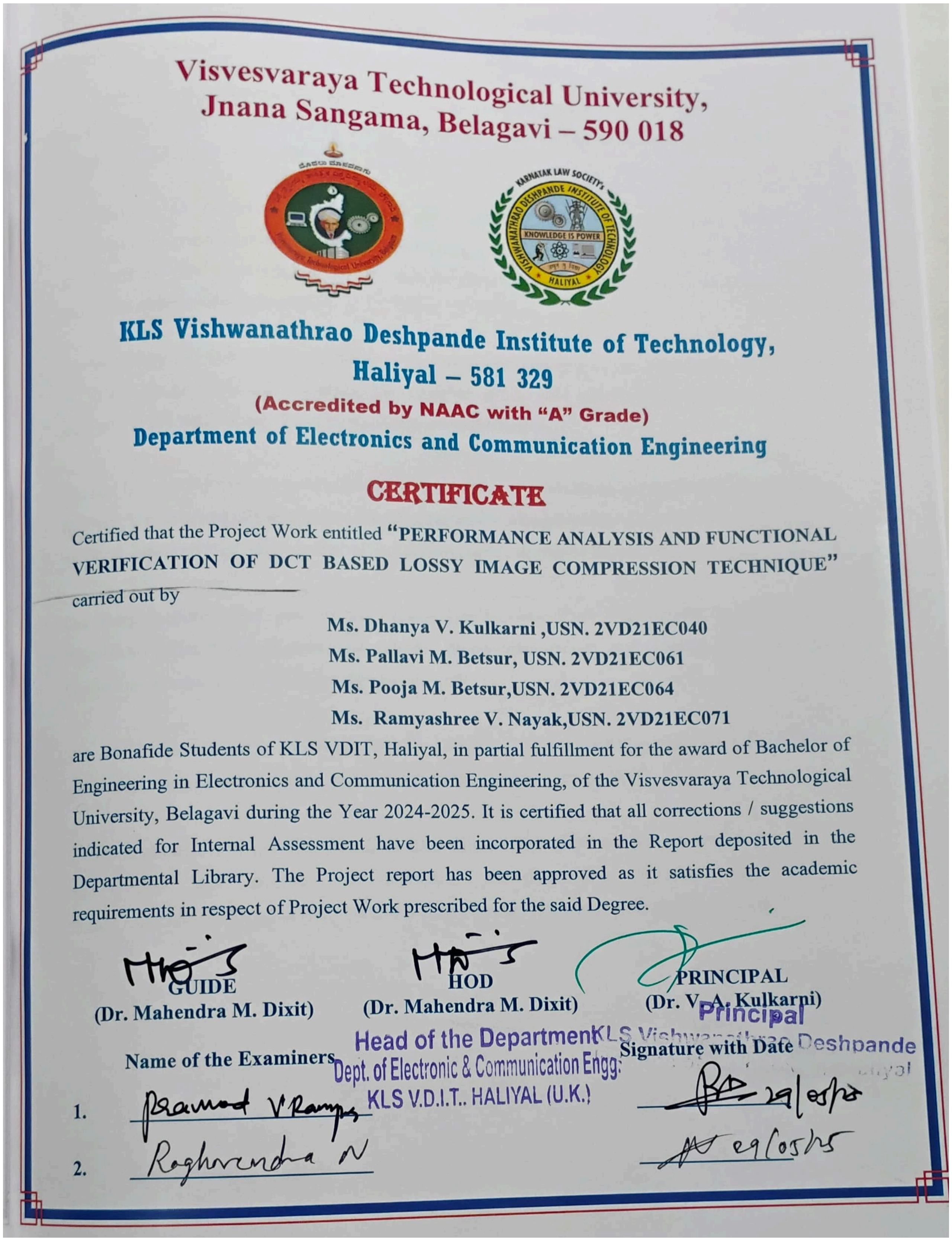
Submitted by

Dhanya .V. Kulkarni	2VD21EC040
Pallavi .M. Betsur	2VD21EC061
Pooja .M. Betsur	2VD21EC064
Ramyashree .V.Nayak	2VD21EC071

VIII – Semester / IV – Year

Under the guidance of

Academic Year 2024– 25 (Scheme 2021)



Visvesvaraya Technological University,
Jnana Sangama, Belagavi – 590 018

KLS Vishwanathrao Deshpande Institute of Technology,
Haliyal – 581 329
(Accredited by NAAC with "A" Grade)
Department of Electronics and Communication Engineering

CERTIFICATE

Certified that the Project Work entitled **"PERFORMANCE ANALYSIS AND FUNCTIONAL VERIFICATION OF DCT BASED LOSSY IMAGE COMPRESSION TECHNIQUE"** carried out by

Ms. Dhanya V. Kulkarni ,USN. 2VD21EC040
Ms. Pallavi M. Betsur, USN. 2VD21EC061
Ms. Pooja M. Betsur,USN. 2VD21EC064
Ms. Ramyashree V. Nayak,USN. 2VD21EC071

are Bonafide Students of KLS VDIT, Haliyal, in partial fulfillment for the award of Bachelor of Engineering in Electronics and Communication Engineering, of the Visvesvaraya Technological University, Belagavi during the Year 2024-2025. It is certified that all corrections / suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the Departmental Library. The Project report has been approved as it satisfies the academic requirements in respect of Project Work prescribed for the said Degree.

[illegible]	[illegible]	[illegible]
GUIDE	**HOD**	**PRINCIPAL**
(Dr. Mahendra M. Dixit)	**(Dr. Mahendra M. Dixit)**	**(Dr. V. A. Kulkarni)**

Head of the Department
Dept. of Electronic & Communication Engg.
KLS V.D.I.T.. HALIYAL (U.K.)

Principal
KLS Vishwanathrao Deshpande
[illegible]

Name of the Examiners		**Signature with Date**
1.	Pramod V Rampur	[illegible] 29/05/25
2.	Raghavendra N	[illegible] 29/05/25

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowded our efforts with success.

We express our deepest gratitude to honorable Principal **Dr. V. A. Kulkarni, KLS's Vishwanathrao Deshpande Institute of Technology, Haliyal** for their constant support and encouragement that went a long way and also for providing all the required resources for the successful completion of our project.

We would also like to extend our gratitude to the Head of the Department **Dr. Mahendra M. Dixit** for his invaluable guidance and assistance. His expertise and support were instrumental in shaping the direction of the seminar and ensuring its smooth implementation.

We feel to acknowledge our indebtedness and deep sense of gratitude to our guide **Dr. Mahendra M. Dixit, Department of Electronics and Communication Engineering** whose valuable guidance and kind supervision given to us throughout the course which shaped the present work.

Our sincere appreciation goes to the Project Coordinators, Prof. Raghavendra N and Prof. Vijaylaxmi Kalal, for their consistent support, timely coordination, and valuable assistance in facilitating the smooth execution of the project work.

We would like to express our sincere gratitude to Ms. G. Suma, our Co-guide, for his valuable guidance, support and encouragement throughout the course of this project.

We thank all faculty members and staff for their support throughout our project, and we gratefully acknowledge the authors of reference materials that enriched our work.
We are especially thankful to our parents for their constant encouragement and inspiration.

Project Associates
Dhanya .V. Kulkarni
Pallavi .M. Betsur
Pooja .M. Betsur
Ramyashree .V.Nayak

ABSTRACT

With the rapid growth of digital imagery and increasing demand for efficient storage and transmission, image compression has become a fundamental aspect of modern multimedia systems. In this project, we present a software-based implementation of image compression using the Discrete Cosine Transform (DCT), a widely adopted technique known for its energy compaction and frequency domain representation capabilities. The proposed method involves preprocessing input images, segmenting them into fixed-size blocks, and applying a 2D-DCT to convert spatial pixel data into frequency components. These coefficients are then quantized using predefined quantization matrices, followed by entropy encoding to achieve substantial data reduction. The model is implemented and tested in MATLAB, employing two quantization strategies—direct and sequential—to analyze trade-offs between compression ratio and image quality. Performance is evaluated using Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE), with results indicating effective compression while maintaining high visual fidelity. This approach demonstrates the practicality and adaptability of DCT-based techniques in software applications and highlights their potential for broader deployment in image storage, web optimization, and multimedia systems.

INDEX

LIST OF FIGURES

Chapter 1:

INTRODUCTION

1.1 Introduction to project and background

In today’s digital landscape, images have become a ubiquitous medium for communication, documentation, entertainment, and data analysis. With the rise of high-resolution imaging in smartphones, surveillance systems, medical diagnostics, and satellite imagery, the volume of image data generated and transmitted has surged exponentially. Consequently, the need for effective image compression and processing techniques is more critical than ever before.

Image compression refers to reducing the amount of data required to represent an image. This is essential for minimizing storage requirements and speeding up transmission over networks without significantly compromising the visual quality. One of the most widely used and fundamental techniques in image compression is the **Discrete Cosine Transform (DCT)**. DCT is a mathematical transform that converts spatial domain data (pixel values) into frequency domain data (coefficients representing image variations).

The key idea behind DCT is **energy compaction**—it transforms an image in such a way that most of its visual information (energy) gets concentrated in a few low-frequency components. This makes it possible to discard high-frequency components (which usually represent fine details or noise) with minimal loss in perceived image quality. This characteristic makes DCT the backbone of the widely-used **JPEG image compression standard**.

In the JPEG algorithm, an image is divided into 8×8 blocks. The DCT is applied to each block, followed by quantization (which reduces less significant frequencies), and the image is then encoded. When decompressed, the image is reconstructed using the inverse DCT (IDCT). This approach results in a significant reduction in file size while maintaining acceptable visual quality—making JPEG efficient for storage and transmission across devices and platforms.

The importance of this project lies in its educational and practical value. By implementing DCT manually or via software tools such as MATLAB one can gain a deeper appreciation for how mathematical concepts translate into real-world multimedia solutions. Additionally, this project provides a simplified foundation upon which more advanced image and video compression systems (like MPEG or HEVC) are built.

1.2 Problem statement

The goal of this project is to implement a software-based image processing system using the Discrete Cosine Transform (DCT) for efficient image compression. The challenge lies in reducing image file size while maintaining visual quality by transforming image data into the frequency domain and eliminating less significant components. This approach is essential for optimizing storage and transmission in applications like multimedia, medical imaging, and surveillance. The project focuses on applying 2D DCT, quantization, and inverse DCT to demonstrate practical image compression and evaluate its effectiveness using PSNR and MSE.

1.3 Objectives

To develop a software-based image processing system that utilizes the Discrete Cosine Transform (DCT) to demonstrate effective image compression and reconstruction. The project aims to transform spatial domain image data into the frequency domain using DCT, allowing less significant high-frequency components to be discarded, thereby reducing file size while maintaining visual quality. It also focuses on implementing quantization and inverse DCT (IDCT) to reconstruct the image and assess the quality using metrics such as Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE). Additionally, the project explores the practical applications of DCT in areas such as denoising, feature extraction, and image storage optimization.

Chapter 2:

LITERATURE SURVEY

1. Title: “Effects of Hybrid SVD–DCT Based Image Compression Scheme”

   Authors: M.M. Dixit, C. Vijaya

   Publication: Springer Lecture Notes in Networks and Systems. Year:2018

   Dixit and Vijaya presented a hybrid image compression algorithm that combines the strengths of Singular Value Decomposition (SVD) and Discrete Cosine Transform (DCT). While DCT efficiently handles energy compaction, SVD excels in dimensionality reduction and maintaining perceptual quality. The authors introduced variable rank matrices and a modified quantization scheme to balance compression ratio and image fidelity. Experimental analysis showed improved PSNR and lower MSE compared to traditional JPEG techniques.

2. Title: “Hybrid DWT-DCT Compression Algorithm with Adaptive RLE”
   Author: Rafea and Salman

   Publication: International Journal of Advanced Computer Science and Applications (IJACSA)

   Year:2018
   This paper introduced an advanced hybrid image compression technique that combines Discrete Wavelet Transform (DWT), Discrete Cosine Transform (DCT), and adaptive Run Length Encoding (RLE) to achieve high compression performance while preserving critical image quality, especially in medical imaging applications. The proposed method leverages the multi-resolution capabilities of DWT and the energy compaction characteristics of DCT to reduce spatial and frequency-domain redundancies effectively.
   In the first stage, the image is transformed using DWT, which breaks it down into different frequency sub bands, separating fine details (high-frequency components) from coarse structures (low-frequency components). This step provides a hierarchical structure suitable for further compression. DCT is then applied to the approximation coefficients from the DWT output, focusing on compacting most of the image’s energy into a small number of coefficients.

3. Title: “Digital Image Ballistics from JPEG Quantization” in 2006.

   Authors: Hany Farid.

   Publication: Dartmouth College Technical Report.

   Year: 2019

   TR No: 2006-583

   This study investigates image ballistics, which is the science of determining the origin and authenticity of digital images. The author presents a framework for identifying the quantization table used in an image and, consequently, the camera or software that generated it. The technique is based on analyzing the specific quantization patterns left by digital cameras or editing tools.

4. Title: “Multi-Level Enhanced Color Image Compression Using SVD and DCT”

   Autor: Garg and Kumar

   Publication: Multimedia Tools and Applications

   Year: 2020

   DOI: 10.4304/jcp.9.3.644-653

   This paper proposes a multi-level hybrid image compression algorithm that effectively integrates Singular Value Decomposition (SVD) and Discrete Cosine Transform (DCT) for compressing color images. The method is designed to strike a balance between compression performance and computational simplicity, making it suitable for practical deployment in systems where processing speed and storage are constrained, such as in embedded vision and surveillance platforms.

   The core idea behind the proposed algorithm is to **decompose image channels (typically RGB)** and process them individually. DCT is first applied to each image block to transform the spatial domain data into frequency components. This step allows for efficient energy compaction—most image information is concentrated in a few low-frequency components. Next, SVD is used to further compress the transformed data by decomposing each frequency matrix into its singular values and vectors, where a reduced number of singular values can be retained to represent the most significant information.

5. Title: “Image Compression by Vector Quantization of DCT Coefficients Using a Self-Organizing Neural Network””

   Authors: Boyapati

   Publication: Proceedings of the International Conference on Information Technology

   Year March 2022.

   This paper presents an innovative image compression algorithm that combines vector quantization of Discrete Cosine Transform (DCT) coefficients with the learning capabilities of a Self-Organizing Neural Network (SOM). The main goal of this hybrid method is to exploit the spatial redundancies in images more intelligently by adapting the quantization process based on image content rather than using static quantization tables, as is done in traditional JPEG compression.

   In this approach, **DCT is used as a preprocessing step** to transform the spatial image data into the frequency domain, where much of the redundant information can be more effectively isolated. The transformed coefficients, which represent image details such as edges, textures, and smooth regions, are then **vector quantized** using a self-organizing neural network. SOMs are unsupervised neural networks that adaptively learn the distribution of input data and form clusters (codebooks) that are representative of the input pattern.

Chapter 3:

METHODOLOGY

3.1 Requirement Analysis and Preliminary Study

The very first stage of the project is crucial as it sets the foundation for the entire development process. It begins with a comprehensive understanding of the problem domain—image processing using Discrete Cosine Transform (DCT). This requires a deep dive into the mathematical theory of DCT, which transforms spatial pixel data into frequency components. DCT is a core technique in image compression standards like JPEG, enabling significant data reduction while maintaining perceptual image quality. The study should include understanding how DCT separates an image into parts of differing importance (frequencies), how it helps in identifying redundancy, and how it is used in compression and reconstruction.

Alongside the theoretical study, it is essential to identify the scope and goals of the software implementation. This includes deciding whether the project will focus on grayscale or color images (color images add complexity due to multiple channels), whether it will support different image formats (such as BMP, PNG, JPEG), and the expected output—whether the goal is just to visualize DCT coefficients, compress and decompress images, or both.

Next, the choice of programming environment plays a pivotal role. MATLAB is also widely used for prototyping image processing algorithms because of its built-in functions and visualization capabilities. C or C++ might be preferred for performance-optimized implementations. The decision depends on the project goals, available resources, and familiarity with the tools.

Additionally, an initial survey of existing methods, libraries, and algorithms related to DCT and image compression can help identify best practices, optimization techniques, and common challenges. This may involve reviewing research papers, textbooks, and open-source projects to ensure that the project leverages proven methods and avoids reinventing the wheel.

The requirement analysis phase also includes defining measurable success criteria such as acceptable image quality metrics after compression (e.g., minimum PSNR thresholds), compression ratio targets, and performance benchmarks. Understanding hardware constraints (such as memory limitations and processing power) and software requirements (such as user interface needs or integration with other systems) can also influence design decisions.

3.2 Image Preprocessing

Before applying DCT, images must be prepared to ensure accurate and efficient processing. This step typically begins by reading the image into the program and converting it to a grayscale format if the focus is on single-channel processing, as this reduces computational complexity while still demonstrating the method's principles. The pixel intensity values may be normalized to a specific range (e.g., 0 to 1) to maintain numerical stability during transformation. The image is then segmented into small, fixed-size blocks, commonly 8x8 pixels, as this size is standard in compression algorithms and balances computational load with compression performance. Segmenting into blocks allows localized frequency analysis and helps to exploit spatial redundancy in the image. Proper handling of image boundaries (e.g., padding) is essential to ensure all pixels are included without errors.

3.3 Implementation of Forward DCT

The forward Discrete Cosine Transform (DCT) is applied to convert image data from the spatial domain into the frequency domain, which helps separate important visual information from less important details. The image, divided into smaller blocks (typically 8x8 pixels), undergoes the 2D DCT individually on each block. This process transforms each pixel's intensity values into frequency coefficients using cosine basis functions. The coefficients near the top-left of the block represent low-frequency components, which contain most of the image's meaningful information, while coefficients towards the bottom-right correspond to high frequencies, which often represent finer details or noise.

Implementing the forward DCT involves computing sums of cosine-weighted pixel values, which can be computationally expensive if done directly. To optimize performance, fast DCT algorithms or existing libraries like OpenCV or MATLAB functions are often used. Special care is taken to handle image sizes not divisible by the block size, typically by padding the image to fit the block structure.

The output of this step is a set of frequency coefficients for each block, which forms the basis for further processing like quantization and compression. This transformation is fundamental to reducing image data redundancy while maintaining visual quality.

3.4 Quantization

The Quantization is the process of reducing the precision of the DCT coefficients to achieve compression by removing less perceptually important information. This is typically done by dividing each DCT coefficient by a corresponding value in a quantization matrix and rounding the result. The quantization matrix is designed so that higher frequency coefficients are more heavily quantized (coarser quantization), reflecting the human eye’s lower sensitivity to high-frequency image details. This step drastically reduces the amount of data needed to represent the image, but introduces some loss of quality. Designing or selecting an appropriate quantization matrix involves balancing compression efficiency against the acceptable level of distortion. Quantization is a key step in lossy image compression.

3.5 Inverse DCT and Image Reconstruction

Once quantization and any further compression are done, the image must be reconstructed to evaluate the quality of the compression. The inverse DCT (IDCT) is applied to each block of quantized coefficients to transform them back from the frequency domain into the spatial domain pixel values. This step effectively reverses the forward DCT, though due to quantization loss, the reconstructed pixel values will not be identical to the original. After inverse transformation, the blocks are combined back to form the complete image. Additional post-processing such as clipping pixel values to ensure they fall within the valid range (0 to 255 for 8-bit images) and correcting any block boundary artifacts is necessary to make the image visually coherent and suitable for display.

3.6 Testing, Validation, and Optimization

The final step involves comprehensive evaluation and refinement of the implementation. Testing is done by applying the software to various images and measuring the quality of the reconstructed images compared to the originals. Objective metrics like Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) quantify the distortion introduced by compression. Visual inspection is also essential to catch artifacts or losses that numeric metrics might miss. Based on testing outcomes, optimization can be performed, such as adjusting quantization parameters, improving algorithm efficiency using faster DCT/IDCT implementations, or experimenting with different block sizes. Profiling tools can help identify bottlenecks in the code to speed up execution. This iterative process ensures that the final software is both effective in compression and efficient in performance.

Chapter 4:

SOFTWARE REQUIREMENTS

- Operating System
- MATLAB
- Required Toolboxes: Image Processing Toolbox
- File Format Support
- ➢ MATLAB supports a variety of image file formats including:

1. .jpg / .jpeg
2. .png
3. .bmp
4. .tif / .tiff

- Code Editor.

Software Prototype Environment

OPERATING SYSTEM:

MATLAB is a cross-platform software and runs smoothly on all major operating systems. For your project, you can use:

- **Windows 10/11**: Offers full compatibility with MATLAB and is widely used in academic and industrial settings.
- **Linux (e.g., Ubuntu 20.04 or later)**: Preferred in research environments and supports scripting and automation well.
- **macOS**: Also fully supported by MATLAB and suitable for academic development.

The choice depends on the system you're comfortable with and the availability of the software license.

MATLAB Software:

MATLAB (Matrix Laboratory) is the main software used in your project. It is a high-level language and interactive environment that makes it easy to perform computational mathematics, visualize data, and develop algorithms.

MATLAB is ideal for image processing projects due to its:

- Strong matrix and array computation capabilities.
- Built-in functions for DCT (dct2, idct2), image I/O, and display.
- Powerful visualization tools to analyze frequency components and image quality.

Required Toolboxes: Image Processing Toolbox:

This toolbox is essential for handling and manipulating images. It provides:

Functions for reading and writing images: imread, imwrite.

- Conversion functions: rgb2gray, im2double
- Image display functions: imshow, imagesc
- Image transformation and analysis functions: dct2, idct2, and filtering operations.

Without this toolbox, many image-related functions required for DCT-based compression would not be available.

File Format Support

MATLAB supports many common image formats, which makes it flexible for working with different types of images:

- **.jpg / .jpeg** – Widely used for web and photographic images; already compressed.
- **.png** – Lossless compression; good for testing image degradation after applying your own compression.
- **.bmp** – Uncompressed format; ideal for raw image data experiments.
- **.tif / .tiff** – High-quality format used in printing and medical imaging; supports multiple channels and bit depths.

These formats can be easily loaded using imread() and saved using imwrite().

Code Editor:

The MATLAB Editor is an integrated part of the MATLAB environment that helps you write, test, and debug code. Key features include:

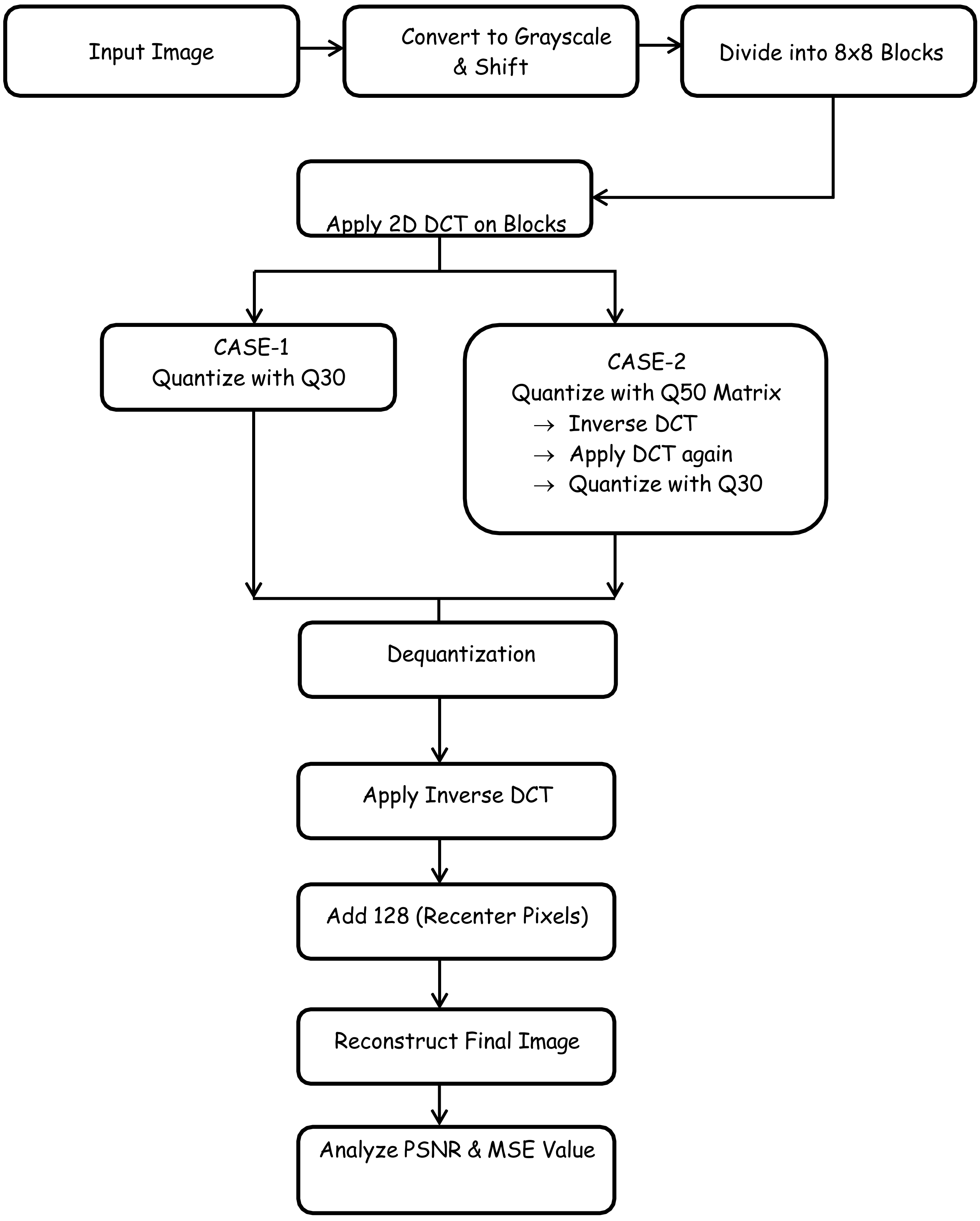
- **Syntax highlighting**: Makes code easier to read and understand.
- **Auto-completion**: Speeds up coding by suggesting functions and variables.
- **Live Scripts (.mlx)**: Combine code, results, text, and images into one file — ideal for documenting your project steps and results.
- **Breakpoints and Debugging**: You can pause execution at specific lines to inspect values and debug step-by-step.
- **Sections (%%)**: Allows breaking the code into logical blocks that can be run independently.
- **Real-time error checking**: Instantly shows issues like missing semicolons or incorrect functions.
- **Git Integration**: Helps with version control if you're tracking changes or collaborating.

This editor is designed to maximize productivity and reduce bugs during algorithm development.

Chapter 5:

IMPLEMENTATION

5.1 Flow chart



Flow Chart Explanation:

1. **Input Image:**
    - This is the source image that will undergo compression.
    - Its purpose is to provide the image on which the DCT-based compression will be applied.
    - A colored or grayscale image is taken as input. If it is colored (RGB), it is later converted to grayscale for DCT processing.

```
image_paths = {'C:\Users\...\flower.jpeg', ...};
num_images = length(image_paths);
```

2. **Convert to Grayscale & Shift:**
    - Converts the color image to a single-channel grayscale image and shifts pixel values**.**
    - DCT works best with grayscale images. Shifting pixel values improves DCT efficiency
    - Annotations help train the machine learning model.
    - The RGB image is converted to grayscale using standard luminance formulas. Then, each pixel value is shifted by subtracting 128, centering values around 0 (range becomes -128 to 127).

```
img_YCbCr = rgb2ycbcr(img);%Convert RGB to YCbCr color space(JPEG standard)

Y = double(img_YCbCr(:,:,1)); % Extract Y (luminance) channel as grayscale

image_normalized = Y - 128;% Shift pixel values from [0,255] to [-128,127]
for DCT
```

3. **Divide into 8×8 Blocks:**
    - The image is segmented into non-overlapping 8×8 pixel blocks.
    - DCT is applied on small blocks to exploit spatial redundancy.
    - The image matrix is broken into 8×8 submatrices, which are processed individually through DCT and quantization steps.

```
blockSize = 8; % Block size for DCT
```

4. **Apply 2D DCT on Blocks:**
    - 2D Discrete Cosine Transform is applied to each 8×8 block.
    - Converts spatial domain data into frequency domain.
    - DCT transforms the pixel values into a matrix of frequency coefficients where most of the important visual information is concentrated in the top-left corner (low frequencies).

```
for i = 1:blockSize:m-blockSize+1
        for j = 1:blockSize:n-blockSize+1
            block = image_normalized(i:i+blockSize-1, j:j+blockSize-1);
            dct_block = dct2(block);
            dct_image(i:i+blockSize-1, j:j+blockSize-1) = dct_block;
            quantized_dct(i:i+blockSize-1, j:j+blockSize-1) = round(dct_block
./ Q30);
        end
    end
```

5. **CASE-1: Quantize with Q30:**
    - Compression by quantizing DCT coefficients using a Q30 matrix.
    - Reduces the number of bits required to represent the image.
    - Each DCT coefficient in the 8×8 block is divided by the corresponding Q30 value and then rounded. Higher values in the matrix lead to greater compression but more quality loss.

```
    % Perform Q30 Compression
    dct_image = zeros(m, n);
    quantized_dct = zeros(m, n);

    for i = 1:blockSize:m-blockSize+1
        for j = 1:blockSize:n-blockSize+1
            block = image_normalized(i:i+blockSize-1, j:j+blockSize-1);
            dct_block = dct2(block);
            dct_image(i:i+blockSize-1, j:j+blockSize-1) = dct_block;
            quantized_dct(i:i+blockSize-1, j:j+blockSize-1) = round(dct_block
./ Q30);
        end
    end
```

6. **CASE-2: Quantize with Q50 Matrix → Inverse DCT → Apply DCT again → Quantize with Q30:**
    - A two-stage compression approach.
    - To analyze how applying quantization twice (Q50 followed by Q30) impacts image quality.
    - First, DCT coefficients are quantized using Q50. Then, inverse DCT reconstructs the image. This image undergoes DCT again, followed by quantization using Q30. This simulates multiple stages of lossy compression.

```
% Perform Q50 Compression
        for i = 1:blockSize:m-blockSize+1
            for j = 1:blockSize:n-blockSize+1
             .  block = image_normalized(i:i+blockSize-1, j:j+blockSize-1);
                dct_block = dct2(block);
                dct_image(i:i+blockSize-1, j:j+blockSize-1) = dct_block;
                quantized_dct(i:i+blockSize-1, j:j+blockSize-1) =
round(dct_block ./ Q50);
            end
        end
```

```
% Now apply Q30 on Reconstructed Q50 Image
        image_normalized = reconstructed_Y_Q50 - 128;
    end

% Perform Q30 Compression
    dct_image = zeros(m, n);
    quantized_dct = zeros(m, n);

    for i = 1:blockSize:m-blockSize+1
        for j = 1:blockSize:n-blockSize+1
            block = image_normalized(i:i+blockSize-1, j:j+blockSize-1);
            dct_block = dct2(block);
            dct_image(i:i+blockSize-1, j:j+blockSize-1) = dct_block;
            quantized_dct(i:i+blockSize-1, j:j+blockSize-1) = round(dct_block
./ Q30);
        end
    end
```

7. **Dequantization:**

    - The inverse of quantization.
    - To approximately retrieve original DCT coefficients.
    - The quantized values are multiplied with the same quantization matrix used earlier (Q30), restoring the scale of the coefficients before inverse DCT.

```
% Dequantize & Apply IDCT (Q50)
        reconstructed_Y_Q50 = zeros(m, n);
        for i = 1:blockSize:m-blockSize+1
            for j = 1:blockSize:n-blockSize+1
                dequantized_block = quantized_dct(i:i+blockSize-1,
j:j+blockSize-1) .* Q50;
                reconstructed_block = idct2(dequantized_block);
                reconstructed_Y_Q50(i:i+blockSize-1, j:j+blockSize-1) =
reconstructed_block;
            end
        end

% Dequantize & Apply IDCT (Q30)
    reconstructed_Y_Q30 = zeros(m, n);
    for i = 1:blockSize:m-blockSize+1
        for j = 1:blockSize:n-blockSize+1
            dequantized_block = quantized_dct(i:i+blockSize-1,
j:j+blockSize-1) .* Q30;
            reconstructed_block = idct2(dequantized_block);
            reconstructed_Y_Q30(i:i+blockSize-1, j:j+blockSize-1) =
reconstructed_block;
        end
    end
```

8. **Apply Inverse DCT:**

    - Performs the Inverse Discrete Cosine Transform on each block.
    - Converts frequency domain data back to spatial domain.

- The dequantized 8×8 blocks are transformed back into image pixel values using inverse DCT.

9. **Add 128 (Restore Pixels):**
   - Restores the original pixel value range.
   - To bring pixel values back to the standard 0–255 range.
   - Adds 128 to each pixel in the block (which was subtracted earlier), shifting values from the range (-128 to 127) back to (0 to 255).
10. **Reconstruct Final Image:**
    - Combines all processed 8×8 blocks into one image.
    - To form the final output image after decompression.
    - The inverse DCT-processed blocks are placed back in their original positions to reconstruct the complete image.

```
% Clip values
    reconstructed_Y_Q30 = uint8(max(min(reconstructed_Y_Q30 + 128, 255),
0));
% Reconstruct final color image by combining Y, Cb, Cr
    reconstructed_YCbCr = cat(3, reconstructed_Y_Q30, Cb, Cr);
    reconstructed_rgb = ycbcr2rgb(reconstructed_YCbCr);
```

11. **Analyze PSNR & MSE Values:**
    - Image quality assessment metrics.
    - To measure the loss of quality after compression.
    - PSNR (Peak Signal-to-Noise Ratio): Indicates how close the compressed image is to the original. Higher PSNR means better quality.
    - MSE (Mean Squared Error): Measures the average squared difference between original and compressed images. Lower MSE indicates better quality.

```
% Compute Metrics for Q30
    mse_Q30 = mean((Y(:) - double(reconstructed_Y_Q30(:))).^2);
    psnr_Q30 = 10 * log10((255^2) / mse_Q30);

% Compute MSE & PSNR for Q50
        mse_Q50 = mean((Y(:) - reconstructed_Y_Q50(:)).^2);
        psnr_Q50 = 10 * log10((255^2) / mse_Q50);
```

» Pseudocode

```
START

Input: Color Image

1. Convert the image to YCbCr color space
2. Extract the Y (luminance) channel for compression
3. Subtract 128 from each pixel in Y to center values around 0

4. Divide the Y channel into 8x8 blocks

5. FOR EACH 8x8 block in the image DO
    a. Apply 2D Discrete Cosine Transform (DCT)
    b. IF User selects CASE 1 (Q50 then Q30)
        i. Quantize using Q50 matrix
        ii. Dequantize using Q50 matrix
        iii. Apply Inverse DCT
        iv. Add 128 to re-center pixel values
        v. Store as intermediate reconstructed image
        vi. Subtract 128 from intermediate image
        vii. Apply DCT again
        viii. Quantize using Q30 matrix
    ELSE IF User selects CASE 2 (Q30 only)
        i. Quantize using Q30 matrix
    END IF
END FOR

6. FOR EACH quantized block DO
    a. Dequantize using Q30 matrix
    b. Apply Inverse DCT
    c. Add 128 to re-center pixel values
END FOR

7. Combine all 8x8 blocks into final Y channel
8. Merge Y with original Cb and Cr channels
9. Convert the image back to RGB color space

10. Compute MSE and PSNR between original and reconstructed images

11. Display:
    - Original image
    - Grayscale (Y) image
    - Reconstructed Y image
    - Final reconstructed color image
    - PSNR and MSE values

END
```

Chapter 6:

RESULTS AND DISCUSSION

Implementing a Discrete Cosine Transform (DCT)- based image compression algorithm is anticipated to yield significant reductions in image data size while maintaining acceptable visual quality. The following outcomes are expected:

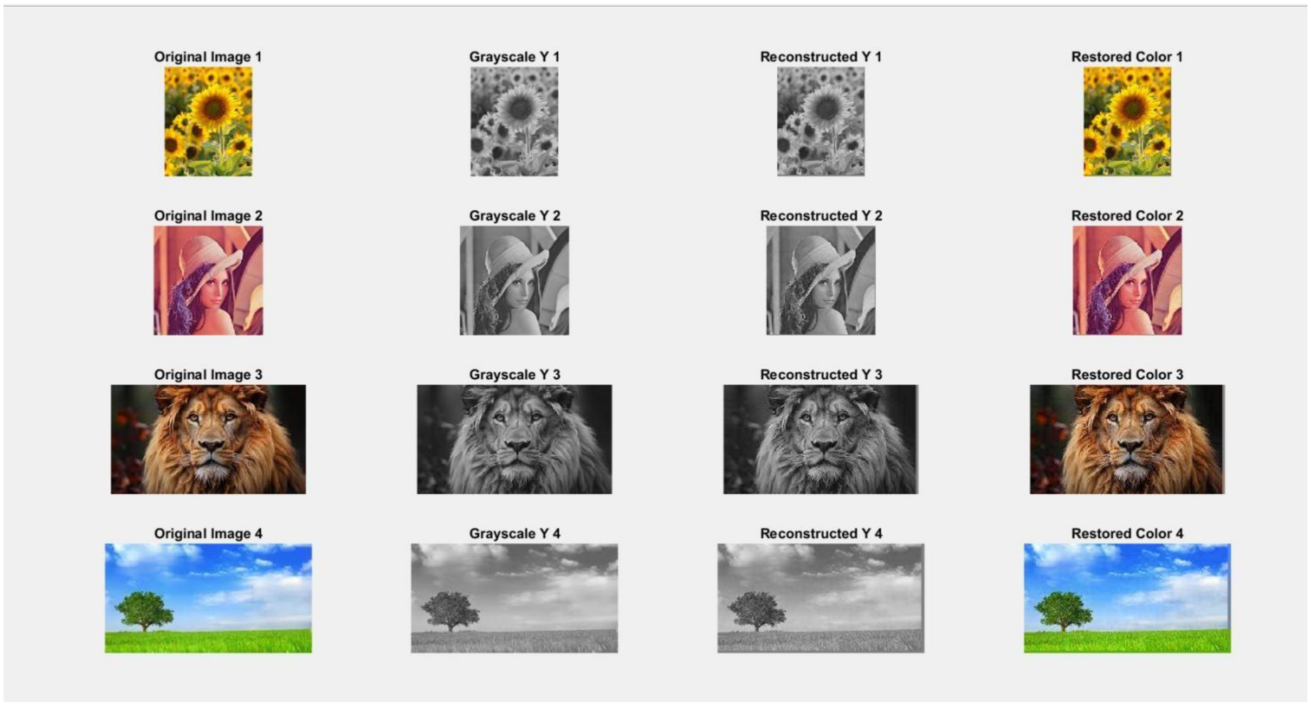


Fig 6.1: Image Processing Results (Original, Grayscale, Reconstructed Y, and Restored Color)

The implemented DCT-based image compression algorithm was evaluated using standard grayscale test images across varying quantization strategies. Two key performance metrics were utilized: **Peak Signal-to-Noise Ratio (PSNR)** and **Mean Squared Error (MSE)**. These metrics provide quantitative insights into the trade-off between compression efficiency and image quality.

A. Quantitative Evaluation Using PSNR and MSE :

Quantitative performance metrics such as Mean Squared Error (MSE) and Peak Signal-to-Noise Ratio (PSNR) provide an objective measure of reconstruction fidelity. Table I presents the computed MSE and PSNR values for each test image compressed with both Q50 and Q30 matrices. It is evident from the data that Q50 compression yields lower MSE values and higher PSNR values across all images. For instance, Image 1 yields a PSNR of 31.56 dB with Q50, compared to 29.33 dB with Q30. Similarly, the MSE increases from 45.41 to 75.94 when switching from Q50 to Q30. These results validate that while Q50 maintains a higher visual accuracy, Q30 incurs greater reconstruction error due to its coarser quantization.

- **Case 1: Direct Q30 Quantization**
  This approach applies a quantization matrix corresponding to a quality factor of 30 directly to the DCT coefficients. The results showed higher compression ratios, but with a moderate decline in image quality. This was evident from lower PSNR values and slightly increased MSE values. Nonetheless, the reconstructed images remained visually acceptable for applications where storage is prioritized over fidelity.

Image	MSE Q50	PSNR Q50 (dB)	MSE Q30	PSNR Q30 (dB)	Compression ratio
1	-	-	61.19	30.26	6.95
2	-	-	80.21	29.09	9.36
3	-	-	176.60	25.66	6.85
4	-	-	71.58	29.58	10.88

Fig 6.2 Table 1: Comparison Table showing MSE and PSNR for Q30

- **Case 2: Sequential Q50 followed by Q30**
  In this method, a two-stage quantization process was used. Initially, the image was compressed with a Q50 matrix and reconstructed, followed by a second DCT and Q30 quantization. This approach aimed to preserve more image details during the first stage, resulting in improved image quality compared to direct Q30 quantization. PSNR values were observed to be higher than in Case 1, and MSE was comparatively lower, indicating better fidelity.

Image	MSE Q50	PSNR Q50 (dB)	MSE Q30	PSNR Q30 (dB)	Compression ratio
1	45.41	31.56	75.94	29.33	5.96
2	65.49	29.97	99.68	28.14	6.87
3	105.67	26.35	195.95	25.21	5.69
4	60.98	30.20	82.68	28.96	9.24

Fig 6.3 Table 2: Comparison Table showing MSE and PSNR for Q50 and Q30

B. Visual Quality Analysis:

A direct visual comparison between original, grayscale, reconstructed, and color-restored images reveals the inherent trade-offs between image quality and compression aggressiveness. As illustrated in Fig. 1, the reconstructed grayscale and color images from Q50 quantization appear sharper and retain finer details of the original images, including textures and contours. In contrast, Q30-quantized reconstructions exhibit noticeable softness and slight blockiness, a common artifact introduced due to heavier compression. However, despite these visible differences, the Q30 results maintain an acceptable level of perceptual quality, particularly when viewed at a standard resolution or in applications tolerant to slight quality degradation, such as web-based image delivery or mobile imaging system.

Chapter 7:

CONCLUSION AND FUTURE SCOPE

This project successfully demonstrates a software-based implementation of Discrete Cosine Transform (DCT) for efficient image compression. By converting spatial domain data into frequency components, the algorithm effectively reduces image redundancy and enables significant compression with minimal perceptual loss. The use of quantization strategies, including both direct and sequential approaches, showcased the balance that can be achieved between compression ratio and image quality.

Experimental results, supported by PSNR and MSE metrics, confirmed that the proposed implementation can achieve high compression ratios while maintaining acceptable visual fidelity. The sequential quantization approach (Q50 followed by Q30) provided a notable improvement in reconstructed image quality over the direct Q30 method, demonstrating the benefit of multi-stage compression strategies.

Overall, the findings reinforce the practicality and efficiency of DCT-based image compression in software environments. This approach holds strong potential for deployment in storage-constrained and bandwidth-sensitive applications, such as web imaging, mobile platforms, and digital archives.

The scope of this work can be broadened through several promising directions, each aimed at enhancing the performance, quality, and applicability of DCT-based image compression systems:

- **Color Image Compression**: While the current implementation emphasizes grayscale processing, future advancements can target full-color image compression using the YCbCr color space. This enables efficient handling of color data by exploiting human visual system characteristics.
- **Hardware Acceleration**: Implementing the DCT and related operations on hardware platforms such as FPGAs or GPUs can drastically reduce computation time. This would allow the system to meet real-time constraints, making it suitable for applications like live video streaming and surveillance.
- **Adaptive Quantization**: The integration of machine learning techniques to generate adaptive quantization matrices offers a dynamic way to optimize compression. This can lead to better preservation of important image regions while still achieving significant size reduction.

Enhancing DCT-based compression can expand its utility across multimedia, medical imaging, and web applications by balancing image quality and data efficiency.

REFERENCES

[1] "Modelling and hardware implementation of quantization levels of digital cameras in DCT based image compression" by M.M. Dixit and C. Vijaya, published in Engineering Science and Technology, an International Journal, 2018 .

[2] Jesse D.kornblum,in: Defense Cyber Crime Institute, United States, Using JPEG quantization tables to identify imagery processed by software, Digital investigation S21-S25, Digital forensic Research Workshop. Published by Elsevier Ltd,2008, https://doi.org/10.1016/j.diin.2008.05.004.

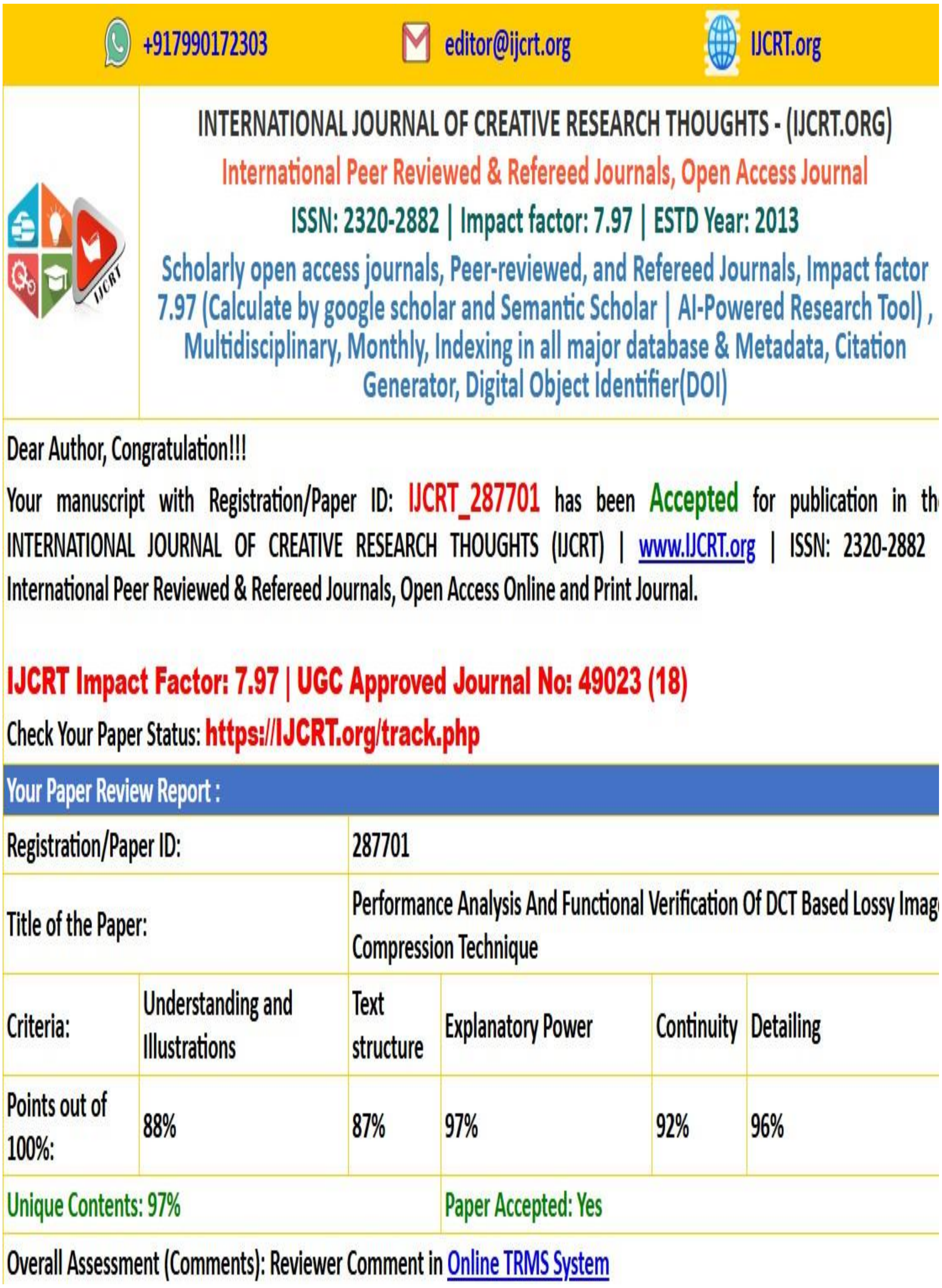
[3] Khalid Sayood, Introduction to Data Compression,3rd Edition, ISBN13:978-0-12620862-7, ISBN 10: 0-12-620862-X, Morgan Kaufmann–Elsevier Publications, 2006.

[4] Mahendra M. Dixit, Priyatam kumar, Comparative Analysis of Variable Quantization DCT and Variable Rank Matrix SVD Algorithms for Image Compression Applications, in: IEEE International Conference on Computational Intelligence and Computing Research, 2010, https://doi.org/10.1109/ICCIC.2010.5705879.

[5] http://www.impulseadventure.com/photo/jpeg-quantization.html.

[6] Min Jeong, Jun Ho Kang, Yong Su Mun, Doo Hee Jung, JPEG Quantization Table Design for Photos with Face in Wireless Handset, 5th Pacific Rim Conference on Multimedia Tokyo, Japan (PCM 2004) Proceedings Part–III, Springer–Verlag Berlin Heidelberg 2004, LNCS 3333, ISBN 3-540-23985-5, pp. 68–688, DOI: http://doi.org/10.1007/978-3-540-30543-9_85.

[7] Hany Farid, Digital Image Ballistics from JPEG Quantization TR 2006– 583, Dartmouth College, Computer Science , 2006, pp. 1–6.



+917990172303 editor@ijcrt.org IJCRT.org

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS - (IJCRT.ORG)

International Peer Reviewed & Refereed Journals, Open Access Journal

ISSN: 2320-2882 | Impact factor: 7.97 | ESTD Year: 2013

Scholarly open access journals, Peer-reviewed, and Refereed Journals, Impact factor 7.97 (Calculate by google scholar and Semantic Scholar | AI-Powered Research Tool) , Multidisciplinary, Monthly, Indexing in all major database & Metadata, Citation Generator, Digital Object Identifier(DOI)

Dear Author, Congratulation!!!

Your manuscript with Registration/Paper ID: **IJCRT_287701** has been **Accepted** for publication in th INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT) | www.IJCRT.org | ISSN: 2320-2882 International Peer Reviewed & Refereed Journals, Open Access Online and Print Journal.

IJCRT Impact Factor: 7.97 | UGC Approved Journal No: 49023 (18)

Check Your Paper Status: **https://IJCRT.org/track.php**

Your Paper Review Report :

Registration/Paper ID:		287701			
Title of the Paper:		Performance Analysis And Functional Verification Of DCT Based Lossy Imag Compression Technique			
Criteria:	Understanding and Illustrations	Text structure	Explanatory Power	Continuity	Detailing
Points out of 100%:	88%	87%	97%	92%	96%
Unique Contents: 97%			Paper Accepted: Yes		

Overall Assessment (Comments): Reviewer Comment in Online TRMS System