JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

An Improved Image Compression Algorithm Using SVD And 2D-DWT

[1]Kiran Basavannappagowda, [2]Jaswanth RS
[1,2] Student
[1,2] MTech in Artificial Intelligence
[1,2]RACE, Reva University, Bengaluru, KA, India

Abstract: In numerous applications, the transmission and storage of images are frequently required. The smaller the image, the lower the associated costs of transmission and storage. Therefore, data compression techniques are often employed to minimize the storage space utilized by the image. One such technique involves applying Singular Value Decomposition (SVD) to the image matrix. In this process, a digital image is subjected to SVD, which decomposes the image into three matrices. Singular values are then utilized to reconstruct the image, resulting in a representation with a reduced set of values, thereby decreasing the required storage space. The objective is to compress the image while retaining the essential features that characterize the original image. SVD is versatile and can be adapted to any arbitrary matrix of m x n size, whether square, reversible, or non-reversible. The compressed image generated by the modified SVD is then used as input for a 2D Discrete Wavelet Transform (DWT) image compression algorithm to achieve a superior compression ratio with the same Mean Square Error (MSE). The performance metrics used are the compression ratio, MSE, Peak Signal-to-Noise Ratio, Bitrate and Structural Similarity Index (SSIM).

Index Terms **- Singular Value Decomposition (SVD); 2D discrete wavelet transform (2D DWT); Inverse Discrete wavelet (IDWT); lossy image compression; Image Performance Metrics; Python; Linear Algebra**

I. INTRODUCTION

With technology playing an increasingly central role in our lives, vast amounts of data are generated daily. This data encompasses diverse formats like text, audio, video, and images. Images, a popular data sharing method, are often used to convey information efficiently ("A picture is worth a thousand words"). The rise of smartphones and smart devices has further facilitated image capture and utilization. As data scientists, we routinely store, process, and transmit enormous volumes of digital data. However, storing uncompressed data incurs significant storage costs, and its transmission demands high bandwidth. Consequently, data compression techniques for storage and transmission have become a crucial research area, particularly in artificial intelligence, pattern recognition, and signal processing. This paper explores the application of Singular Value Decomposition (SVD) for reducing storage requirements of image files.

Image Compression: Balancing Quality and Efficiency: Image compression aims to minimize the data needed to represent images. Since images are digital, the goal is to represent them using the fewest possible bits. Redundancies within the image file, defined as repetitive patterns contributing to image resolution, can be exploited to achieve this reduction. However, the process should not excessively compromise image quality, rendering it unusable. A good image compression algorithm strikes a balance between these two competing factors. The application typically dictates the optimal trade-off between compression and data quality.

Lossless vs. Lossy Compression Techniques: Image compression techniques fall into two broad categories: lossless and lossy. Lossless compression defines entropy, a theoretical limit for data reduction. It aims to produce a bit-by-bit replica of the original data. This reversible process preserves image quality. In contrast, lossy compression identifies and eliminates minute details and variations in the image that the human eye may not readily perceive. This approach significantly reduces storage requirements by discarding such features, but often at the cost of some image quality degradation. Lossy compression is irreversible, meaning the discarded information cannot be recovered. However, it allows for much higher compression ratios compared to lossless techniques.

Image Representation: Pixels and Matrices: Images are typically represented using matrices with dimensions m x n. Here, m represents the number of rows, corresponding to the image height in pixels, and n represents the number of columns, corresponding to the image width in pixels. Each element (pixel) within the matrix contributes to the overall image representation. Pixel intensity, relative to surrounding pixels, is denoted by a numerical value assigned to each pixel. These values determine the characteristics of the corresponding pixel. In grayscale images, values range from 0 (black) to 1 (white), reflecting the relative greyscale level of each pixel. Colored images, composed of red, green, and blue (RGB) channels, are further divided by the computer into these three layers during storage. This additional layer necessitates more storage space for colored images compared to grayscale images.

Consequently, each pixel in a colored image is associated with three values ranging from 0 (no color) to 1 (fully saturated), representing the red, green, and blue intensity levels.

II. BACKGROUND

Data redundancy significantly impacts image compression. This redundancy falls into three categories:

Coding redundancy: Suboptimal code usage leads to inefficiencies. Techniques like Huffman coding address this.

Inter-pixel redundancy: Neighboring pixels exhibit correlation, allowing for value prediction. This redundancy is also known as spatial or geometric redundancy.

Psycho-visual redundancy: The human eye has varying sensitivity to different frequencies. Techniques exploit this for targeted compression without compromising perceived quality.

SVD and 2D -DWT implementations leverage these redundancies to achieve smaller image sizes by efficiently eliminating redundant information

Singular Value Decomposition (SVD):

Singular Value Decomposition (SVD) is a powerful technique for dimensionality reduction and data analysis. It aims to represent a high-dimensional dataset using a lower number of dimensions while capturing the most important information.

Here's a breakdown of how SVD works:

Dimensionality Reduction: SVD takes a high-dimensional data matrix and decomposes it into a set of lower-dimensional matrices. This process reveals the underlying structure of the data by identifying the most significant variations within it. Think of it as simplifying a complex dataset by focusing on the most informative aspects.

Matrix Factorization: SVD essentially factorizes the original data matrix (M) into three component matrices:

U: An orthogonal matrix of size m x m (where m is the number of rows in the original matrix).

Σ (Sigma): A diagonal matrix of size m x n (where n is the number of columns in the original matrix). This diagonal matrix contains the singular values, which are the square roots of the eigenvalues obtained during the decomposition process. Singular values represent the importance of the corresponding basis vectors in the decomposition.

VT: An orthogonal matrix of size n x n, representing the transpose of another orthogonal matrix V.

The relationship between these matrices is expressed as: $M = U\Sigma V^T$.

Eigenvectors and Singular Values: SVD relies on the concept of eigenvalues and eigenvectors. The decomposition involves calculating the eigenvalues and eigenvectors of AAT (where A is the original data matrix) and ATA. These eigenvalues and eigenvectors are then used to construct the U and V matrices, respectively. The singular values are obtained from the eigenvalues by taking the square root. The eigenvalues and, consequently, the singular values are arranged in descending order along the diagonal of the Σ matrix. This ordering signifies the importance of each dimension (basis vector) in representing the data.

In essence, SVD provides a compact representation of the original data by identifying the most significant variations and discarding less important details. This makes it a valuable tool for image compression and other dimensionality reduction tasks.

Applying SVD in image compression:

As discussed earlier, SVD decomposes a matrix into a sum of simpler rank-one matrices. When applied to an image matrix, SVD breaks it down into three component matrices. However, simply applying SVD doesn't achieve compression.

The key to compression lies in exploiting the singular values obtained after SVD decomposition. These singular values are arranged in descending order along the diagonal of the singular value matrix (Σ). The first singular value holds the most significant information about the image, while subsequent values contribute progressively less. Lower singular values, therefore, represent negligible image details.

Discarding these lower singular values during reconstruction allows us to reduce the image size without introducing significant distortion to the original image. Here's a formalized representation of the process:

Let A be the matrix representing the image. SVD decomposes it as:

$A = U \Sigma V^T = \sum(\mu_i * u_i * v_i^T)$ (summation from 1 to r)

Expanding the summation:

$A = \mu_1 u_1 v_1^T + \mu_2 u_2 v_2^T + ... + \mu_r u_r v_r^T$

We can truncate this summation by discarding terms with smaller singular values (μ_i) before reconstruction. This results in a compressed version of the image matrix:

$A_k = \mu_1 u_1 v_1^T + \mu_2 u_2 v_2^T + ... + \mu_k u_k v_k^T$

The storage requirement for this compressed matrix (A_k) is k(m + n + 1) units, where m and n are the dimensions of the original image matrix. The value of k, which controls the level of compression, will be close to but less than n. This ensures a good balance between image resolution and compression ratio. By adjusting k, we can tailor the compression level to specific storage requirements.

In essence, SVD-based compression leverages the inherent redundancy in image data by discarding less significant information captured by lower singular values. This allows for efficient storage and transmission of images while maintaining an acceptable level of visual fidelity. This process is illustrated in below image.

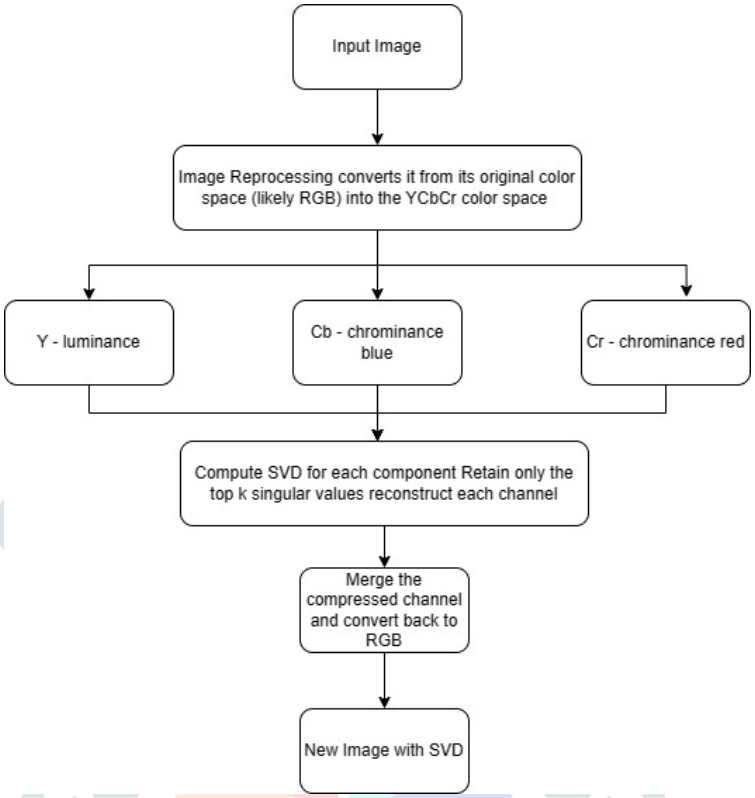


Figure 1. Image compression using SVD method

2 D - discrete wavelet transform for Image Compression:

2D Discrete Wavelet Transform (DWT) is a powerful technique for image compression. It decomposes an image into different frequency sub-bands, allowing us to store or transmit the most important information while discarding less crucial details. Here's a breakdown of the process with matrices:

1. Image as a Matrix:

   Imagine your image as a grayscale picture. We represent this image as a 2D matrix I where each element I[i,j] represents the intensity of the pixel at row i and column j. This matrix essentially captures the spatial information of the image.

2. Filter Application:

   DWT uses filters to decompose the image. We have two sets of filters:
   Low-pass filter (H): Captures the overall trends and smooth areas of the image.
   High-pass filters (V and D): Capture horizontal and diagonal details like edges and textures.
   These filters are small matrices themselves, typically of size 2x2.

3. Decomposition Steps:

   Here's how we decompose the image using these filters and matrices:

   a. Row-wise filtering:

      * Apply the low-pass filter (H) to each row of I to obtain the approximation component (LL). This captures the overall structure of the image.
      * Apply the high-pass filters (V and D) to each row to get horizontal detail (HL) and diagonal detail (LH) components. These capture sharp changes in intensity.

   b. Column-wise filtering:

      * Take the filtered rows obtained in step (a) and treat them as a new matrix.
      * Apply the same set of filters (H, V, D) to the columns of this matrix.

4. Resulting Sub-bands:

   After these steps, we obtain four sub-bands represented by a matrix structure:
   LL (Approximation): This sub-band contains the most significant information about the image, representing its overall structure.
   HL, LH, HH (Details): These sub-bands capture horizontal, vertical, and diagonal details like edges and textures, respectively.

5. Compression with Quantization:

For compression, we typically focus on the LL sub-band since it holds the most crucial information. We can then:
Quantize the LL sub-band (reduce the precision of its values) while maintaining acceptable image quality.
Discard the HL, LH, and HH sub-bands entirely for high compression ratios (lossy compression), leading to a smaller representation of the image but with some loss of detail.

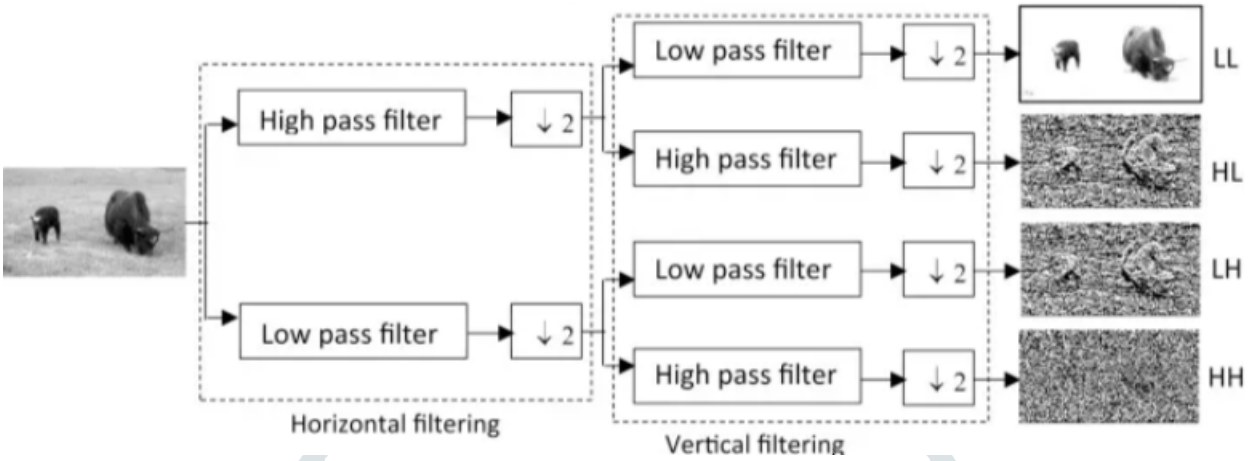


Figure 2. 2D discrete wavelet transform

Inverse Discrete wavelet transform for Image Compression:

Inverse Discrete Wavelet Transform (IDWT) for image reconstruction involves the following steps:

1. Up sampling:
   Up sample each wavelet component (LL, LH, HL, HH) to match the size of the original image by inserting zeros between each element in each row and column.

2. Reconstruction Along Columns:
   - Apply synthesis filters along the columns of each unsampled component.
   - Convolve each column of the unsampled components with the synthesis filters and sum the results to reconstruct the columns of the original image.

3. Reconstruction Along Rows:
   - Apply synthesis filters along the rows of the reconstructed columns.
   - Convolve each row of the reconstructed columns with the synthesis filters and sum the results to reconstruct the rows of the original image.

Synthesis filters, comprising a low-pass synthesis filter (LPSF) for the approximation component (LL) and a high-pass synthesis filter (HPSF) for the detail components (LH, HL, HH), are essential for the reconstruction process. These filters are designed to achieve perfect reconstruction, ensuring that the original image can be exactly reconstructed from its wavelet coefficients. The synthesis filters play a crucial role in combining the unsampled components to reconstruct the original image accurately.

The reconstruction process involves applying the synthesis filters to the unsampled components and summing the results to obtain the final reconstructed image. This process ensures that the original image is faithfully reconstructed from its wavelet components, completing the inverse wavelet transform.

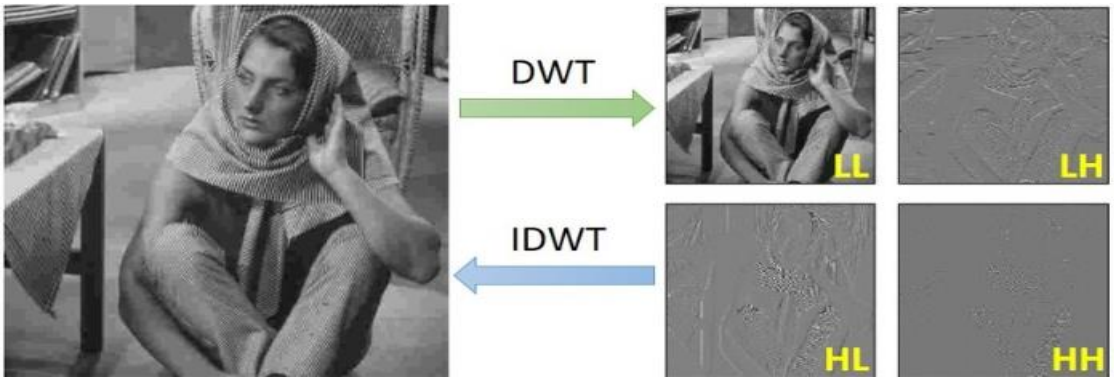


Figure 3. Image Compression using 2D-DWT and IDWT

III. PROPOSED SOLUTION

The image matrix is composed of RGB signals. However, for the purposes of storage and transmission, the RGB representation is not optimal due to the presence of redundant information. The proposed solution seeks to address this by transforming the RGB spectrum into the YCBCR spectrum. Here, Y represents the brightness or luminance, while CB and CR represent the blue and red chroma components, respectively. This new representation approximates the process where primary colors are transformed into perceptually meaningful information. It segregates the Y components into two Chroma components, which can be stored at a higher resolution or transmitted at a higher bandwidth. They can also be treated separately, either by compression or bandwidth reduction.

The proposed system involves using the YCBCR format to represent the original image, which is in RGB representation. The constituent components can be obtained by breaking down the YCBCR representation, yielding three components corresponding to Y, CB, and CR. Singular Value Decomposition (SVD) is then applied to these components to obtain their respective U, Σ, and V components, namely, Uy, Σy, Vy corresponding to Y, Ub, Σb, Vb corresponding to CB, and Ur, Σr, Vr corresponding to CR. Since U and V represent the spatial attributes of the image, they can be taken from any color space. However, since Y represents the brightness, it must be used for all three components. This approach can achieve information loss. Since Y is used in all three components, we have chosen to use U and V of the Y component. All three components of the image are used in reconstruction to reduce the damage due to loss. Frequency components for U and V are generated, and a thresholding technique is used to remove the low-value singular values from the matrix. This method helps to achieve a significant amount of compression while retaining the quality. The image is reconstructed using Σ, U, and V of all three components Y, CB, and CR. Compression is enhanced by applying the compressed images as input to the 2D Discrete Wavelet Transform (DWT) algorithm.

Performance metrics are evaluated to compare the original and compressed images. These include the compression ratio, Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), Bitrate, and Structural Similarity Index (SSIM).

Implemented Algorithm:

Algorithm Input: Image
Algorithm Output: Compressed image

Steps:

1. Convert the RGB image into the YCBCR color space.
2. Decompose the YCBCR image matrix into its constituent components: Y, CB, and CR.
3. Apply the Singular Value Decomposition (SVD) algorithm to each component, generating three vectors for each.
4. Reconstruct the intermediate image by applying the reconstruction algorithm to the decomposed components.
5. Extract the frequency components for the U and V components.
6. Utilize threshold values from the U and V matrices to reconstruct the resultant image.
7. Concatenate the decomposed components Y, CB, and CR to form the final image.
8. Enhance compression by applying the compressed images as input to the 2D Discrete Wavelet Transform (DWT) algorithm.
9. Reconstruct the image from LL, HL, LH, HH components
10. Evaluate performance metrics to compare original and compressed images: compression ratio, Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), Bitrate, and Structural Similarity Index (SSIM)

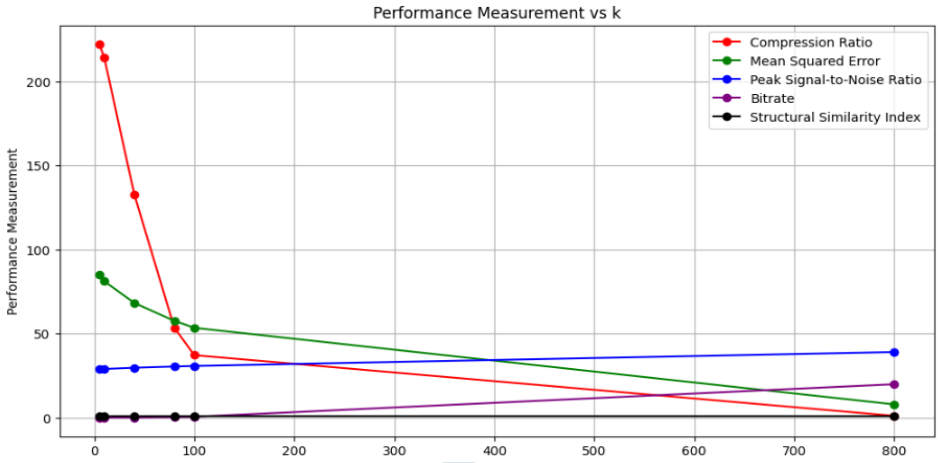
IV. RESULTS

The algorithm was implemented in Python and evaluated on a 139KB image. We iteratively processed the image using the algorithm, varying the number of retained columns (k). For each iteration, the output image size and resolution were measured.

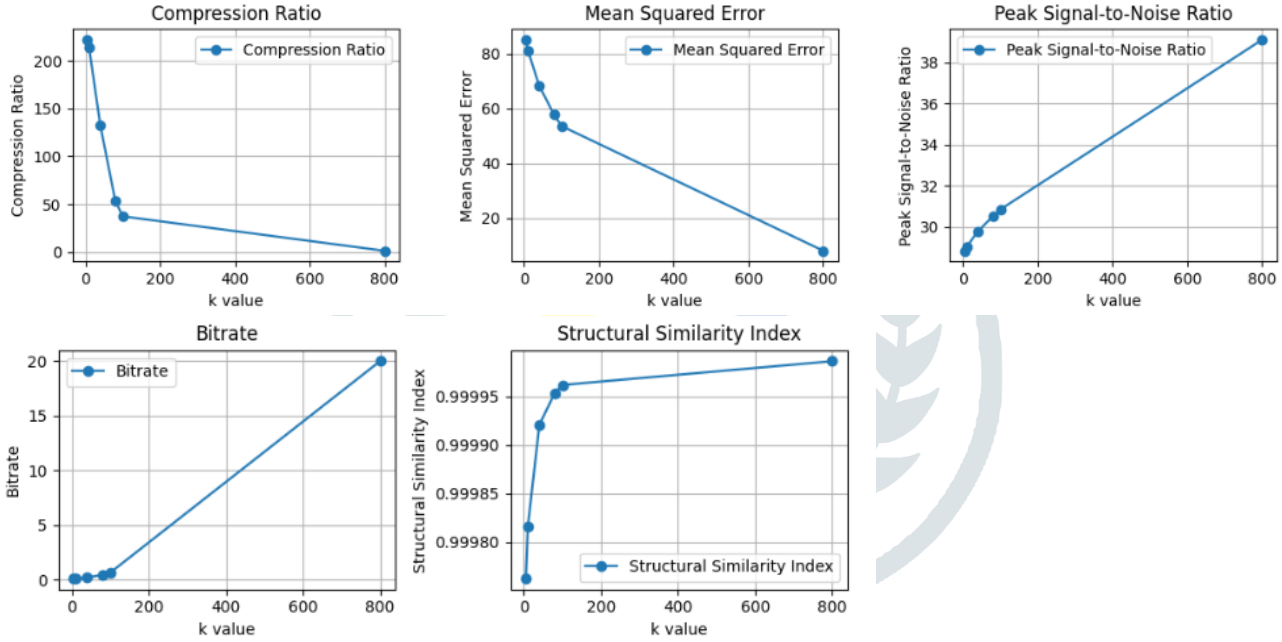
Parameter(K)	Original Image Size (KB)	Compressed Image Size (KB)	Compression Ratio	Mean Squared Error (MSE)	Peak Signal-to-Noise Ratio (PSNR) (dB)	Bitrate (bpp)	Structural Similarity Index (SSIM)
5	139.09	0.63	222.2	85.03	28.84	0.11	0.9997626
10	139.09	0.65	213.86	81.16	29.04	0.11	0.999816
40	139.09	1.05	132.74	68.29	29.79	0.18	0.9999207
80	139.09	2.6	53.54	57.66	30.52	0.45	0.9999534
100	139.09	3.73	37.27	53.51	30.85	0.64	0.9999616
800	139.09	115.87	1.2	8.01	39.1	19.99	0.999986

Table 1. Compressed image performance metrics corresponding to different values of k

The graph to show performance of a compression algorithm on an image at different number columns (K).



The graphs to show various performance metrics of a compression algorithm on an image at different number columns (K).



Performance metrics:

Here we considered below metrics as evaluation criteria.

Compression Ratio (CR):

The Compression Ratio (CR) measures the efficiency of a compression algorithm. It is defined as the ratio between the size of the original image data and the size of the compressed image data , typically expressed in bits. Mathematically, CR is calculated as:

CR = original image size in bytes / Compressed image size in bytes

MSE (Mean Squared Error):

This metric measures the average squared difference between corresponding pixels in the original and compressed images.

Equation: MSE = (1 / m x y) * Σ Σ (f(x, y) - F(x, y))^2

- m x y: Total number of pixels in the image.
- Σ Σ: Double summation symbol, iterating over all pixels in the image.
- f(x, y): Pixel value in the original image at position (x, y).
- F(x, y): Pixel value in the compressed image at position (x, y).

Bitrate (BPP):

The Bitrate (BPP) indicates the average number of bits used to represent each pixel in the compressed image. For color images, BPP can be calculated as 24 / CR, while for grayscale images, it's 8 / CR.

Peak Signal-to-Noise Ratio (PSNR):

The Peak Signal-to-Noise Ratio (PSNR) is a widely used metric for assessing the quality of a compressed image compared to the original. Typically used for 8-bit images, PSNR is expressed in decibels (dB) and calculated using the following formula:

PSNR(dB) = 10 * log10 (255^2 / MSE)

Here, 255 represents the maximum possible value for an image pixel. MSE (Mean Squared Error) refers to the average squared difference between corresponding pixels in the original and compressed images. It is calculated as:

MSE = (1 / m x y) * Σ Σ (f(x, y) - F(x, y))^2

where:

m x y represents the total number of pixels in the image

f(x, y) represents the pixel value in the original image at position (x, y)

F(x, y) represents the pixel value in the compressed image at position (x, y)

Structural Similarity Index (SSIM):

The Structural Similarity Index (SSIM) is a metric that goes beyond simple pixel-wise comparison and assesses the structural similarity between two images [40]. It considers three key aspects:

Luminance: This compares the brightness of corresponding pixels in the original and compressed images.
Contrast: This compares the local variations in pixel intensity between the two images.
Structure: This compares the underlying patterns or textures present in the images.

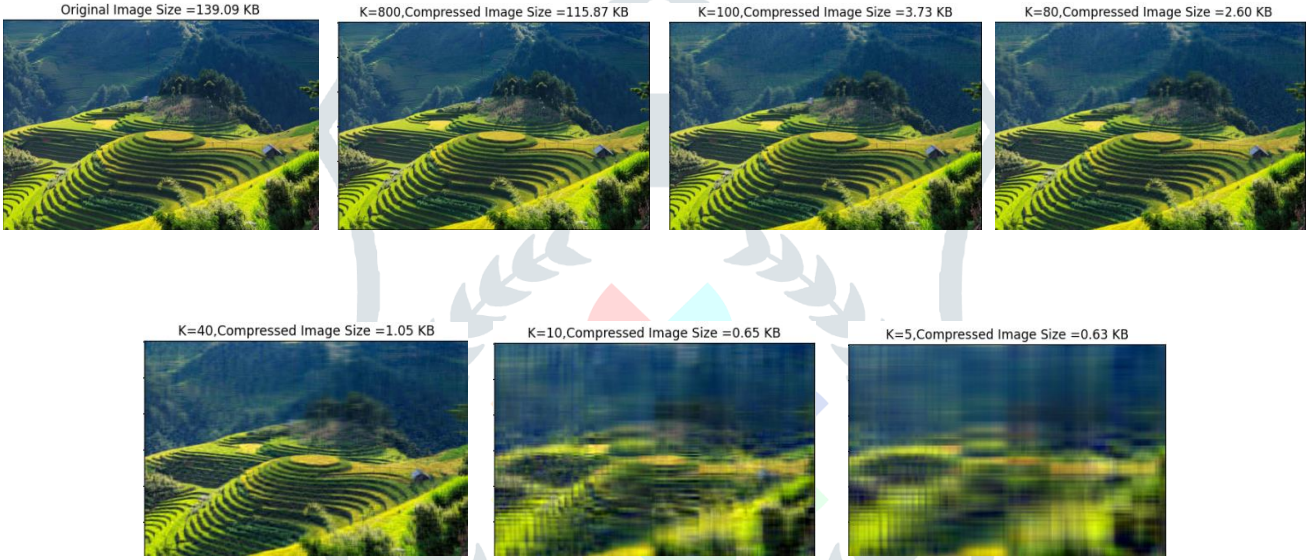
The SSIM score ranges from -1 to 1, with 1 indicating identical images.

Here's the formula for SSIM:

SSIM(x, y) = l(x, y) * c(x, y) * s(x, y)
where:
x represents the original image
y represents the compressed image
l(x, y), c(x, y), and s(x, y) represent the luminance, contrast, and structure comparison functions, respectively.

These metrics provide valuable insights into the trade-off between compression efficiency (measured by CR and BPP) and image quality (measured by PSNR and SSIM) achieved by a lossy compression technique.

Compression of images with different values of k.



V. Conclusion

Based on the presented results, the selection of the parameter k allows for tailoring the compression ratio to specific application requirements. This highlights a key advantage of the proposed SVD-based approach. As the rank (k) of the decomposed matrices increases, a larger portion of the image information is retained in the top singular values. This observation aligns with the inherent property of SVD, where the most significant information is concentrated in the leading singular value coefficients.

Furthermore, the reconstructed matrix size grows with increasing rank (k). This allows for a gradual recovery of image detail as the number of singular values used for reconstruction increases.

The combination of SVD with 2D DWT and IDWT step offers an additional benefit: reduced computational complexity. SVD itself is a relatively efficient decomposition technique, and the inclusion of 2D DWT and IDWT can potentially further enhance compression without incurring a significant computational burden. This characteristic makes the proposed method suitable for real-time or resource-constrained image compression applications.

VI. Acknowledgment

We would like to thank REVA University and JB Shima for their guidance and support throughout this research project

References

[1] H R Swathi, Shah Sohini, Surbhi and G Gopichand (2017)Image compression using singular value decomposition
[2] Rajiv Ranjan and Prabhat Kumar (2023)An Improved Image Compression Algorithm Using 2D DWT and PCA with Canonical Huffman Encoding