A COMPARATIVE ANALYSIS OF IMAGE COMPRESSION TECHNIQUES: K MEANS CLUSTERING AND SINGULAR VALUE DECOMPOSITION

A PROJECT REPORT

Submitted by

AISHWARYA JAYARAMAN [Reg No:RA1911026010102]
ANSHUMAN PAL [Reg No: RA1911026010117]

Under the guidance of

Dr.C. AMUTHA DEVI

(Associate Professor, Department of Computer Science and Engineering)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M. Nagar, Kattankulathur, Kancheepuram District

NOVEMBER 2021

SRM Institute of Science and Technology

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report titled "**A COMPARATIVE ANALYSIS OF IMAGE COMPRESSION TECHNIQUES: K MEANS CLUSTERING AND SINGULAR VALUE DECOMPOSITION**" is the bonafide work of "**AISHWARYA JAYARAMAN [Reg No: RA1911026010102] & ANSHUMAN PAL [Reg No: RA1911026010117]**", who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was con-ferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr.C. AMUTHA DEVI
GUIDE
Associate Professor
Dept. of Computer Science and Engineering

SIGNATURE

Dr.B. AMUTHA
HEAD OF THE DEPARTMENT
Dept. of Computer Science and Engineering

Signature of the Internal Examiner

Signature of the External Examiner

ABSTRACT

The global drive to digitize almost all the existing processes has mandated the conversion of all concerned analog data into their respective digital formats. One such crucial data that is being digitized on a priority in today's world is image. Image Compression is also useful for improving customer engagement and search rankings, sending and uploading images, for storage management etc. Image compression can be achieved through several algorithms.

In this project, we will be comparing two algorithms: K-Means Clustering & Singular value Decomposition for image compression.

K-Means algorithm is a centroid based clustering technique which clusters the dataset into k different clusters, each cluster represented by its centroid point. The pixels of the data points are replaced by the pixels of their respective centroid points, thus giving us the compressed image.

Singular value decomposition algorithm is a matrix factorization technique which finds the best approximation of original pixel data points that are of huge dimensions, with fewer dimensions. It removes redundant data by performing low rank approximation. The most crucial data are stored in certain singular values. So, we can eliminate some of the singular values for bringing about better compression.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my guide, Dr.C. AMUTHA DEVI for her valuable guidance, consistent encouragement, personal caring, timely help and providing us with an excellent atmosphere for doing research. All through the work, in spite of her busy schedule, she has extended cheerful and cordial support to us for completing this research work.

TABLE OF CONTENTS

ABBREVIATIONS

SVD	Singular Value Decomposition
PSNR	Peak Signal to Noise Ratio
CR	Compression Ratio
SSIM	Structural Similarity Index

CHAPTER 1

INTRODUCTION

1.1 Image and Image Compression:

An image is a two-dimensional array of pixels where each pixel is a fundamental unit which can represent color on its own. Each pixel has its own value which in the case of RGB color space has values ranging from 0 to 255.

Image compression is an application of data compression that encodes the original image with a few bits. The objective of image compression is to reduce the redundancy of the image and to store or transmit data in an efficient form. Compressing images is a neat way to shrink the size of an image while maintaining the resolution.

There are two types of image compression, Lossy and Lossless.

Lossy Compression: When the image is compressed at the cost of its clarity and quality, it is known as Lossy compression. The original image data which was worked upon is lost due to inexact approximations and partial data discarding. Lossy compression is often used to compress multimedia data, especially in applications such as streaming media.

Lossless Compression: When an image is compressed without the compromise in image data and the quality of the image is always retained, then the technique is known as Lossless compression. Lossless compression is used in cases when its important that the original image and the decompressed data be identical to one another. Lossless compression is often used in applications like ZIP formats and the GNU tools gzip. It's also used in MP3 encoders.

1.2 Image Compression Techniques:

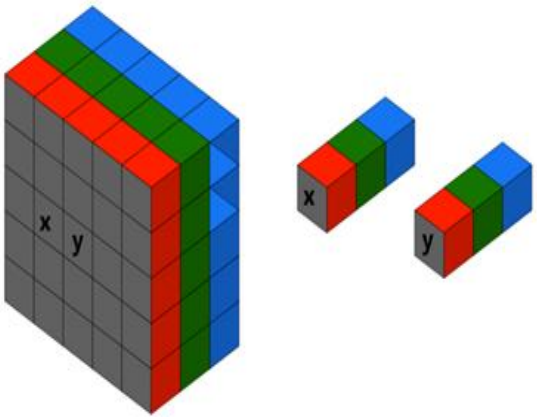
1.2.1 K - Means Clustering

K-means clustering, is an unsupervised machine learning algorithm, which divides the data points into k predefined clusters based on the minimal distance from the centroid. The algorithm is initialized by selecting a random set of values, around which many other values are grouped. Then all the values in the cluster are replaced based on the centroid which is near it. Iterative process of these steps will reduce the number of colors used in the image. As a result, the image is compressed successfully.

In a colored image, each pixel is of size 3 bytes (RGB), where each color can have intensity values from 0 to 255. Following combinatorics, the total number of colors which can be represented is 256*256*256. We can visualize only a few colors in an image. Colors having different values of intensity that are RGB values seem the same to the human eye. The K-Means Clustering algorithm takes advantage of the visual perception of the human eye, uses few colors to represent the image and clubs similar looking colors (which are close together in a cluster).

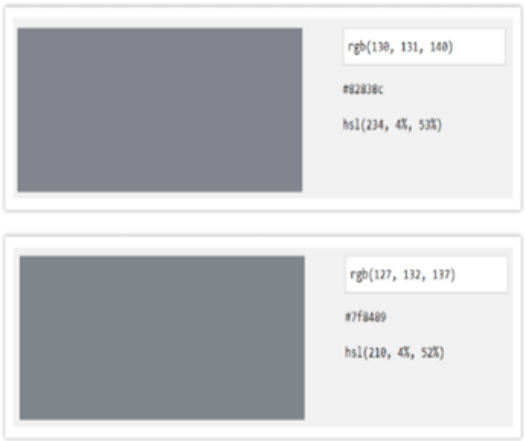
Here's an illustration of how this works:

As shown in the image below, few pixels are picked and expanded in the further images to continue the illustration.



The pixels picked from the previous image are expanded in the given image and two nearby pixels are picked (named as ‘x’ and ‘y’).

If the RGB value of ‘x’ and ‘y’ pixel is (130, 131, 140) and (127, 132,137) respectively, then below is the illustration of how these two-pixel colors are visible.



In the image, it is observed that for some amount of changes in the RGB values the color resembles the same to a human eye. So K-Means clustering can club these two colors together and represent it by a centroid point that has almost the same resemblance to a human eye.

We will be using k-Means clustering to find the k number of colors which will be representative of its similar colors. These k-colors will be centroid points from the algorithm. Then we will replace each pixel value with its centroid points. The color combination formed using only k values will be very less compared to the total color combination, hence reducing the image size. We try different values of k and observe the output image.

Algorithm -

Step 1: Read the image (input) as an array.

Step 2: Fix the value for K and number of iterations.

Step 3: Initialize the algorithm by selecting K number of centroids.

Step 4: Compute the distance and assign the pixel data points to the nearest cluster to which the centroid belongs.

Step 5: Update the centroids by computing the mean of data points in the cluster.

Step 6: Repeat step 4 and step 5 till there is convergence in centroids or until the number of iterations are completed.

Step 7: Construct the image using the array which has undergone the clustering algorithm.

1.2.2 Singular Value Decomposition

Singular Value Decomposition, is an unsupervised machine learning algorithm, which decomposes a matrix into a product of a square matrix, a diagonal matrix and another square matrix.

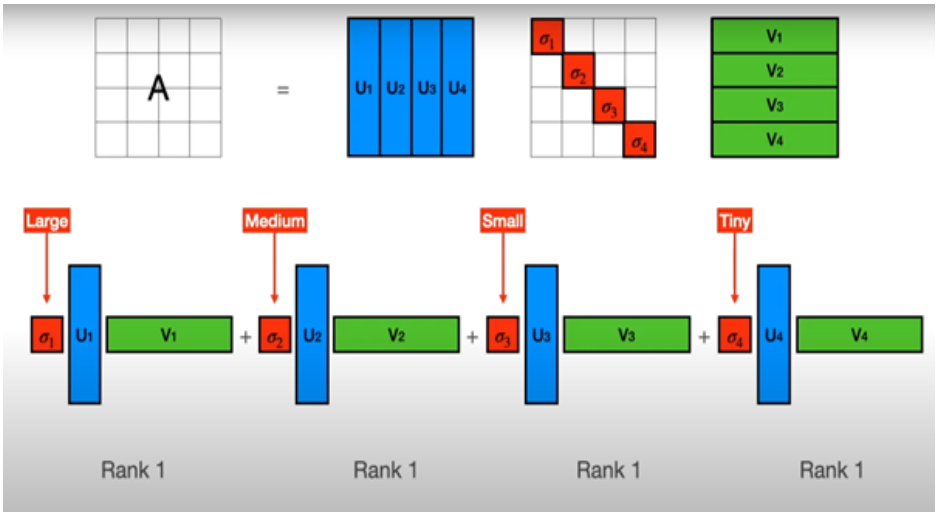
Any m x n image can be represented as m x n matrix. Pixel values of the image are the elements of the m x n matrix.

The algorithm involves breaking down the image matrix A into the form $A = U S V^T$.

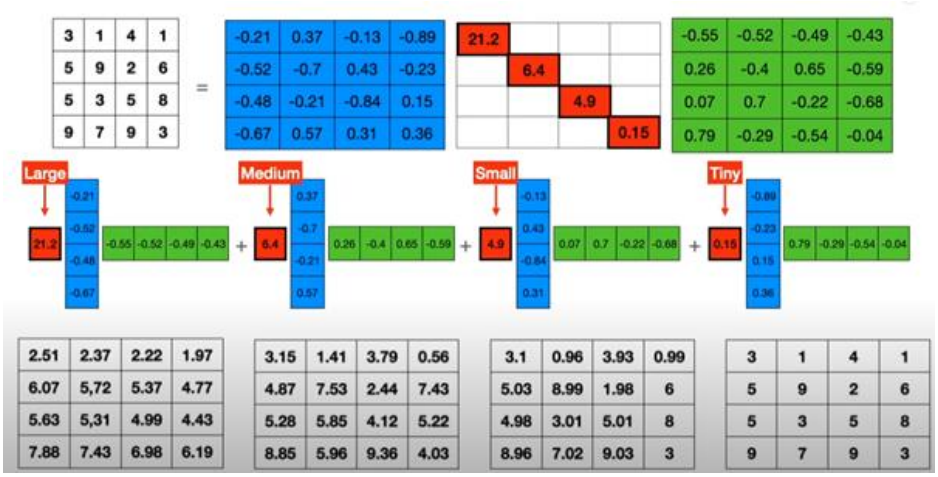
U (orthogonal) is a m x m matrix containing orthonormal Eigenvectors of AAT , S is an m x n matrix consisting of singular values which are the positive square root of non-zero Eigenvalues of ATA and V(orthogonal) is an n x n matrix containing the orthonormal Eigenvectors of A TA.

This computation allows us to retain the important singular values that the image requires while releasing the values that are not as necessary in retaining

the quality of the image.



It removes redundant data by performing low rank approximation.



The most crucial data are stored in certain singular values. So, we can eliminate some of the singular values for bringing about better compression.

Algorithm -

Step 1: Read the image (input) as an array.

Step 2: Fix the value for K and Number of iterations.

Step 3: Isolate the red, green and blue channels of the image.

Step 4: Perform SVD on each channel.

Step 5: Merge all the channels together to reconstruct the image.

CHAPTER 2

LITERATURE SURVEY

Table 2.1: Research Papers

S. No.	Research Paper
1.	Image Compression methods based on transform coding and fractal coding
2.	A Comparative analysis of image compression techniques: K Means Clustering and Singular Value Decomposition
3.	Analysis and Study Of K-Means Clustering Algorithm
4.	Image Compression using Singular Value Decomposition
5.	An Improved SVD based Image Compression

2.1 Image Compression methods based on transform coding and fractal coding

2.1.1 Abstract

In this research paper comparative analysis of image compression is done by four transform methods, which are Discrete Cosine Transform (DCT), Discrete Wavelet Transform(DWT) & Hybrid (DCT+DWT) Transform and fractal coding. Comparison of various Image compression techniques for different images is done based on parameters, compression ratio (CR), mean square error (MSE), peak signal to noise ratio (PSNR).

2.1.2 Discrete Cosine Transform (DCT)

Advantages:

It requires less computational resources and can achieve the energy compaction property.

It has very strong energy compaction, i.e. its large amount of information is

stored in a very low frequency component of a signal and other frequencies having very small data which can be stored by using very less number of bits (usually, at most 2 or 3 bit).

It gives a good compromise between information packing ability and computational complexity.

Disadvantages:

For the higher compression ratio, it introduces the blocking artefacts and the false contouring effects while image reconstruction.

Only spatial correlation of the pixels inside the single 2-D block is considered and the correlation from the pixels of the neighbouring blocks is neglected.

2.1.3 Discrete Wavelet Transform (DWT)

Advantages:

It has the capacity of multi resolution compression.

It provides high quality compression at low bit rates.

It avoids blocking artefacts which degrade reconstructed images.

Disadvantages:

It provides lower quality than JPEG at low compression rates.

It requires longer compression time.

2.1.4 Hybrid (DCT + DWT)

Advantages:

It reduces blocking artefacts, false contouring and ringing effect.

It gives a higher compression ratio as compared to DCT and DWT.

Disadvantages:

It requires more computational time.

It has a higher loss of information.

2.1.5 Fractal Coding

Advantages:

It has a fast decoding process and High compression ratio and works well with low performance devices.

It is resolution independent and can be digitally scaled to any resolution when

decoded.

It has lower transmission time.

Disadvantages:

It has a long encoding time.

Use of fixed size blocks for the range and domain images. There are regions in images that are more difficult to code than others.

2.1.6 Conclusion and Future Work

DWT gives a better compression ratio without losing more information about the image, but requires more processing power. DCT overcomes this disadvantage but gives less compression ratio. DCT based standard JPEG uses blocks of image, but block boundaries are noticeable in some cases. Blocking artefacts can be seen at low bit rates. In a wavelet, there is no need to divide the image. Hybrid transform gives higher compression ratio but for getting that clarity of the image is partially a trade off. Fractal Image Compression gives a great improvement on the encoding and decoding time. A weakness of the proposed design is the use of fixed size blocks for the range and domain images. There are regions in images that are more difficult to code than others. Therefore, there should be a mechanism to adapt the block size (R, D) depending on the mean and variance calculated when coding the block.

The result in this, provides a strong foundation for future work for the hardware design. The algorithm can be realized in hardware implementation as a future work. It can also be a good option for the image processor of the wireless capsule endoscopic system. The research work has been analysed for high compression ratio. Further research can be performed to relax high compression ratio constraints. This work has been constrained only for the removal of the spatial redundancy by compression of still images.

2.2 A Comparative analysis of image compression techniques: K Means Clustering and Singular Value Decomposition

2.2.1 Abstract

In this paper, they discussed two such algorithms, namely K- means clustering and Singular value decomposition. K Means Clustering technique helps in minimizing the colour components of the image. Singular Value Decomposition technique can be carried out by low rank approximation of the image matrix. This research work is performed in Python and the efficiency of both the methods is compared. The comparative analysis of the simulation results is further compared with the existing methods to show the competence of different methodologies.

2.2.2 K-Means Clustering

K-means clustering divides the data points into k predefined clusters based on the minimal distance from the centroid.

The algorithm is initialized by selecting a random set of values, around which many other values are grouped. Then all the values in the cluster are replaced based on the centroid which is near it. Iterative process of these steps will reduce the number of colors used in the image. As a result, the image is compressed successfully.

2.2.3 Singular Value Decomposition

Singular Value Decomposition decomposes a matrix into a product of a square matrix, a diagonal matrix and another square matrix. Any m x n image can be represented as m x n matrix. Pixel values of the image are the elements of the m x n matrix.

SVD involves breaking down the image matrix A into the form A = USVT

This computation allows us to retain the important singular values that the image requires while releasing the values that are not as necessary in retaining the quality of the image.

2.2.4 Conclusion

The two algorithms were compared based on the following parameters –

a) Peak Signal to Noise Ratio (PSNR):

It calculates the ratio between the maximum possible value of a signal and the power of distorting noise that affects the quality of its representation (in decibels). It is used as a quality measurement between the original image and a compressed image. Higher the PSNR value, better the image quality.

b) Structural Similarity Index (SSIM):

It quantifies image quality degradation that is caused by processing such as data compression or data transmission.

c) Compression ratio (CR):

The ratio between original image size and compressed image size. Higher the compression rate, better the output results.

If the K value is smaller, the compression rate in both the algorithms are higher. By analyzing and comparing these two algorithms we can see that the output is better in the K Means Clustering Algorithm with lower K value.

These techniques can be used for both personal and business purposes. Comparative analysis of both the two considered algorithms with other existing techniques shall help to identify the efficient method.

2.3 Analysis And Study Of K-Means Clustering Algorithm

2.3.1 Abstract

In this paper, the behaviour of K means clustering is studied. Through this paper they have tried to overcome the limitations of the K-means algorithm. The actual K-mean algorithm takes a lot of time when it is applied on a large database, so the proposed clustering concept comes into picture to provide a quick and efficient clustering technique on a large data set. In this paper performance evaluation is done for a proposed algorithm using Max Hospital Diabetic Patient Dataset.

2.3.2 Algorithm Proposed

Proposed cluster algorithm (D, Tth)

1. Let k=1
2. // Randomly choose a object from D, let it be p
   k1= {p}
3. K={k1}
4. c1=p
5. C={c1}
6. Assign a constant value to Tth
7. for l=2 to n do
8. Choose the next random point from D other than already choose points let it be q.
9. Determine m, distance between q and centroid
   cm(1<=m<=k) in C such that distance is minimum.
10. If (distance<=Tth) then
11. km=km union q
12. Calculate new mean (centroid cm) for cluster km.
13. Else k=k+1

14. kk={q}

15. K=K union {kk}

16. ck=q

17. C=C union {ck}

2.3.3 Advantages of proposed Clustering Algorithm over Existing K-Means Clustering Algorithm:

In K-means clustering algorithms, the number of clusters (k) needs to be determined beforehand but the proposed clustering algorithm generates the number of clusters automatically.

K-means depends upon initial selection of cluster points, it is susceptible to a local optimum and may miss global optimum. Proposed clustering algorithm is employed to improve the chances of finding the global optimum.

K-means is sensitive to outliers since a small number of outliers can substantially influence the mean value. In proposed clustering algorithm outliers can't influence the mean value and can be easily identified and removed.

Only the cluster representations i.e. centroid are stored permanently in main memory to alleviate space limitations thus space requirements of the proposed algorithm are very small, necessary only for the centroids of clusters. In K-means memory space is more required to store each object permanently in memory along with centroids.

2.3.4 Conclusion and Further Work

The proposed algorithm and the experimental result demonstrate that the given algorithm can improve the direct K-means algorithm.

As threshold value decreases Square Error decreases. Lower the value of Square Error, higher the compactness of the cluster and as separate as possible.

As we decrease the threshold value, cluster quality increases.

As we decrease the threshold value, the number of clusters formed increases.

As we decrease the threshold value number of objects as Outlier increases.

Table1 Comparison of algorithm's running time

Name of algorithm	Worst case	Average case	Best case
k-means	$O(n^i)$ where $2<=i<3$	$O(n^2)$	$O(n)$
Proposed Algorithm	$O(n^2)$	$O(n^i)$ where $1<=i<=2$	$O(n)$

There are several improvements possible to the basic strategy presented in this paper. One approach will be to use the concept of Nearest Neighbor Clustering Algorithm to improve the compactness of clusters.

2.4 Image Compression using Singular Value Decomposition

2.4.1 Abstract

This paper presents an image compression technique, Singular Value Decomposition. Basic mathematics of SVD is dealt with in detail and results of applying SVD on an image are also discussed. The MSE and compression ratio are used as thresholding parameters for reconstruction. SVD is applied on a variety of images for experimentation.

2.4.2 SVD Image Compression

The input image is represented as the m x n matrix. The image reconstructed using SVD technique will reduce the storage space requirement to k*(m+n+1) bytes as against the storage space requirement of m*n bytes of the original uncompressed image.

K is the number of Eigenvalues for compression.

This implies that value of k should be smaller than m*n / (m+n+1) in order to compress any image thus putting an upper limit on the value of k. In short, the value of k is chosen such that a good amount of compression is achieved while image quality is maintained above the minimum acceptable limit.

2.4.3 Conclusion

Following conclusions were drawn –

Smaller the value of k, the more is the compression ratio (i.e. less storage space is required) but image quality deteriorates (i.e. larger MSE and smaller PSNR values).

Thus, it is necessary to strike a balance between storage space required and image quality for good image compression. The optimum compression results are obtained when MSE of the compressed image is just less than or equal to 30dB (i.e. MSE $\leq$ 30dB). In our case, this is obtained when the value of k is

128.

The choice of k depends on the application. For instance, in some applications, if image quality is important then higher values of k are chosen. But sometimes storage space is more important than image quality, in that case lower k values are taken.

When k is equal to the rank of the image matrix, the reconstructed image is almost the same as the original one. And as k is increased further, there is a very negligible decrease in MSE and increase in PSNR values. This means that there is very negligible improvement in the image quality.

SVD finds application in noise reduction, face recognition, watermarking, etc.

2.5 An Improved SVD based Image Compression

2.5.1 Abstract

The research paper presents an improved approach over the technique using the orthonormal property of the matrices produced in SVD decomposition method.

2.5.2 Proposed SVD Image Compression Technique

SVD transform decomposes the image A of size □ × □ into 3 matrices U (matrix of size □ × □ containing eigenvectors of ATA), S (diagonal matrix of size □ × □ containing singular values in decreasing order) and V (matrix of size □ × □ containing eigenvectors of AA^T). Hence the overall size increases with that of originally being □ × □ to □ × □ + □ × □ + □ (□ < □). Hence the compression ratio may be defined as (□ × □) / (□ × □ + □ × □ + □) for the k principal components. For compression to be effective, this ratio must be greater than 1.

$$\frac{m \times n}{k \times (m+n+1)} > 1 \ or \ k < \frac{m \times n}{m+n+1}$$

This expression is traditionally being used for the SVD based compression ratio.

PROPOSED WORK –

Since the eigenvectors are orthogonal and unitary, we do not need to store all the values of the eigenvectors. If we make use of the orthogonal property of the eigenvectors, we can recover the values even if for the ith vector, we skip i-1 values and store other values of the eigenvectors. For the first vector, we store all the values and hence we skip the upper triangular values of the eigenvector's matrices U and V.

The new compression ratio is –

$$CR' = \frac{m \times n}{k(m+n+1) - k(k-1)} = \frac{m \times n}{k(m+n-k+2)}$$

Now since we are skipping some values there must be some gain in the newly formed expression for compression ratio as compared to the previous expression.

$$Gain = CR' - CR$$

$$Gain = \frac{m \times n}{k(m+n-k+2)} - \frac{m \times n}{k(m+n+1)}$$

2.5.3 Conclusion and Future Work

They have used Copydays original images dataset for calculating performance metrics.

Compression ratio increases by the new technique while PSNR and SSIM remain the same.

Gain in compression ratio is higher as the rank increases while at rank 1, it is zero i.e. compression ratio is same for both techniques.

In this paper, a modification to the SVD based image compression is proposed. The modified expressions are derived and by experiments shown to be better than that of the previous technique. The results can be interpreted as showing the natural trade-off between selected rank, k and compression ratio but with greater efficiency, while keeping the complexity same. The results indicate that the proposed scheme improves the compression ratio while maintaining the image quality.

This technique can be further extended to identifying similar blocks and skipping them and using vector coding for the eigenvectors to deliver much more efficient performance

CHAPTER 3

WORK-FLOW DIAGRAM

3.1 K-MEANS CLUSTERING

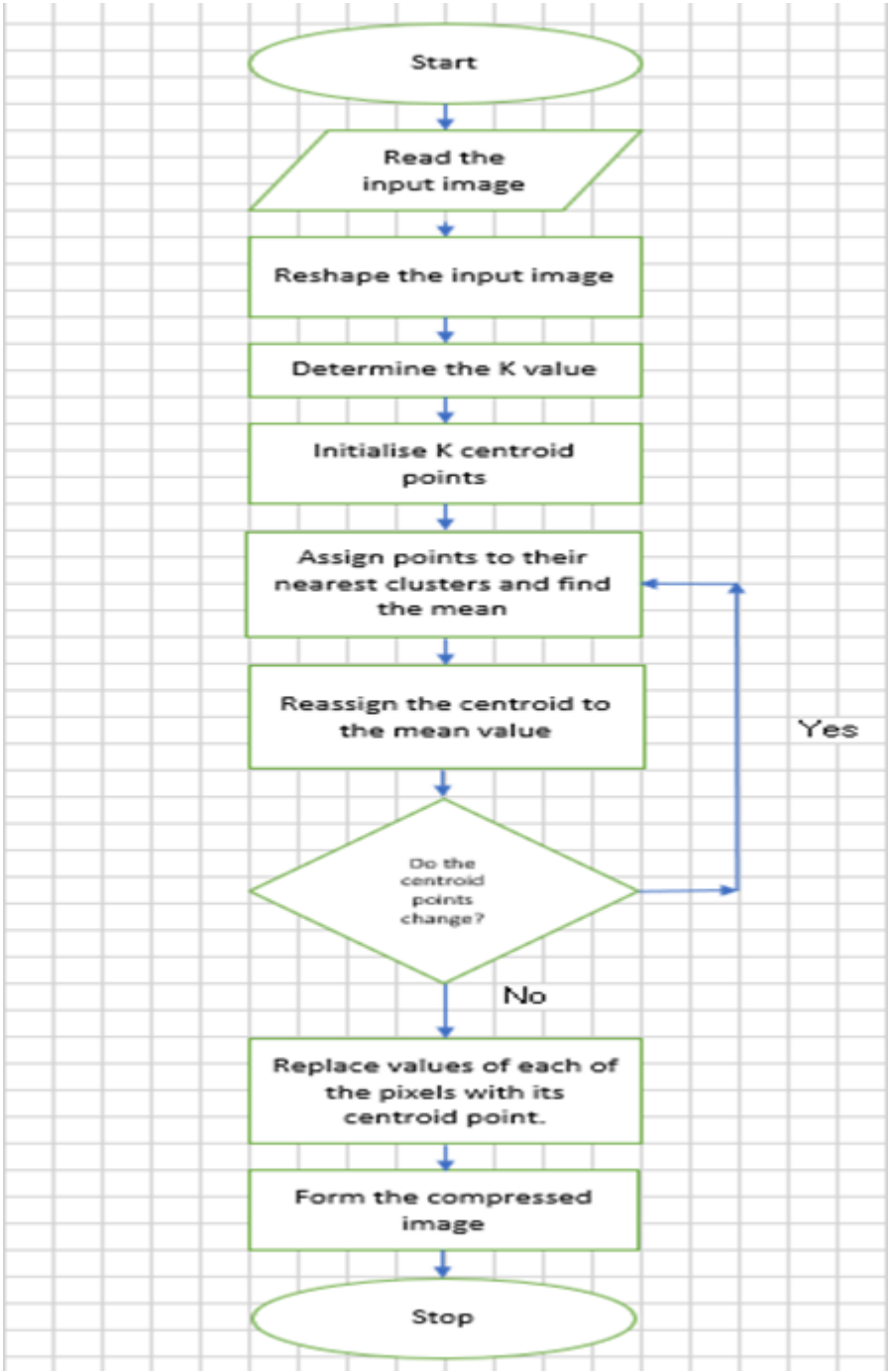


Figure 3.1: K - Means Clustering

3.2 SINGULAR VALUE DECOMPOSITION

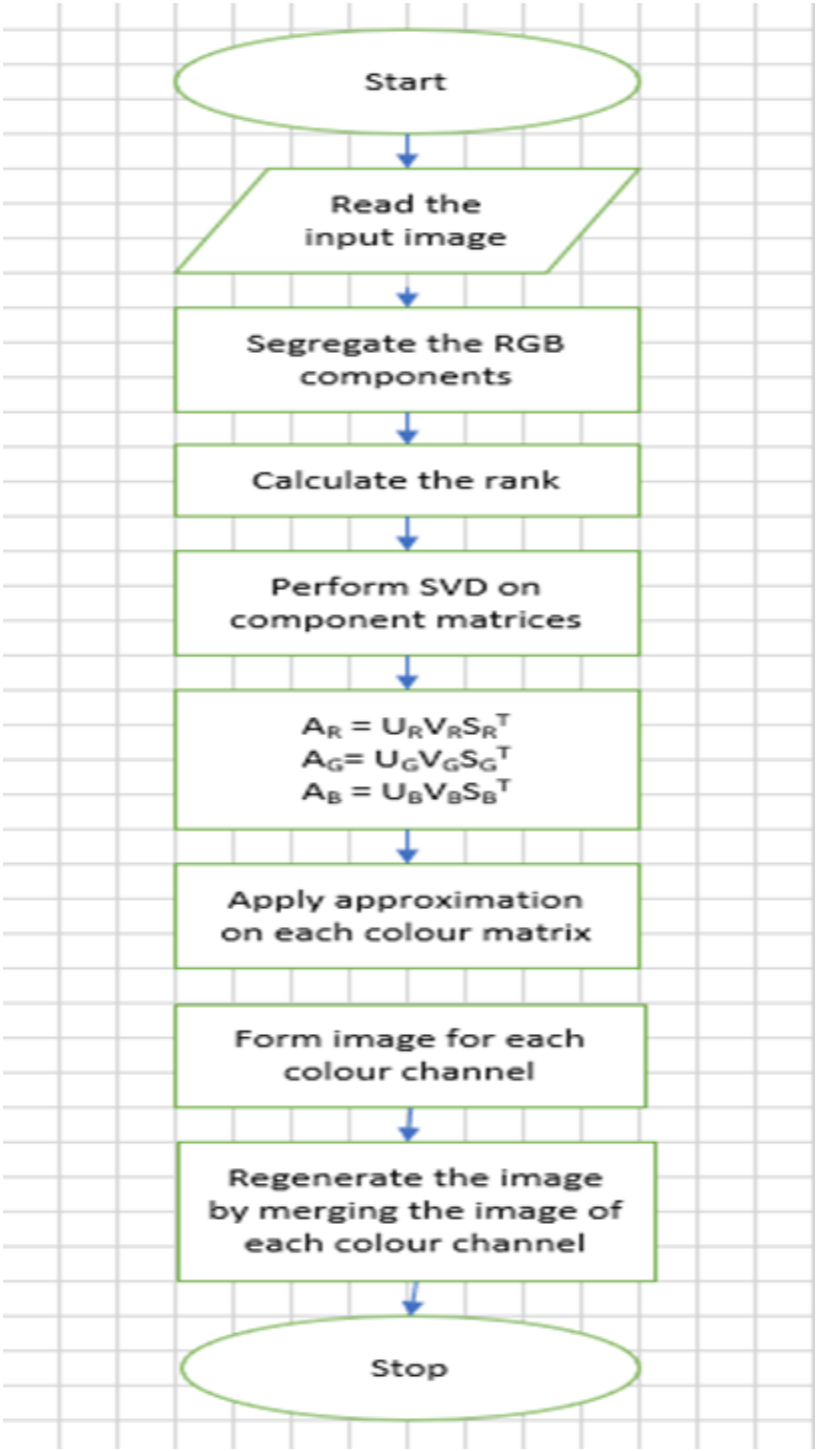


Figure 3.2: SVD

CHAPTER 4

MODULE DESCRIPTION

4.1 K - Means Clustering

4.1.1 Data Pre-Processing

Load the image from the disk. The function returns the image as a NumPy array.

The size of the input image is (rows, cols, 3), flatten all the pixel values to a single dimension of size (rows*cols) and the dimension of each pixel is 3 representing RGB values. The size of the flatten image will be (rows*cols, 3).

```
def load_image(path):
    """ Load image from path. Return a numpy array """
    image = Image.open(path)
    return np.asarray(image) / 255

# Load the image
image = load_image("dog.png")
w, h, d = image.shape
print('Image found with width: {}, height: {}, depth: {}'.format(w, h, d))

# Get the feature matrix X

X = image.reshape((w * h, d))
```

4.1.2 Clustering

Initialize K centroid points. Take a random point and assign it to the cluster with the nearest centroid, compute the mean and reposition the centroid to the mean value. Repeat this process till the centroids aren't moving anymore.

```
def initialize_K_centroids(X, K):
    """ Choose K points from X at random """
    m = len(X)
    return X[np.random.choice(m, K, replace=False), :]


def find_closest_centroids(X, centroids):
    m = len(X)
    c = np.zeros(m)

    for i in range(m):
        # Find distances
        distances = np.linalg.norm(X[i] - centroids, axis=1)

        # Assign closest cluster to c[i]
        c[i] = np.argmin(distances)
    return c


def compute_means(X, idx, K):
    _, n = X.shape
    centroids = np.zeros((K, n))
    for k in range(K):
        examples = X[np.where(idx == k)]
        mean = [np.mean(column) for column in examples.T]
        centroids[k] = mean
    return centroids


def find_k_means(X, K, max_iters=10):
    centroids = initialize_K_centroids(X, K)
    previous_centroids = centroids
    for _ in range(max_iters):
        idx = find_closest_centroids(X, centroids)
        centroids = compute_means(X, idx, K)
        if (previous_centroids==centroids).all():
            # The centroids aren't moving anymore.
            return centroids
        else:
            previous_centroids = centroids
    return centroids, idx

K = 60 # the number of colors in the image

# Get colors
print('Runnign K-means')
colors, _ = find_k_means(X, K, max_iters=20)
```

Because the indexes returned by the find_k_means function are 1 iteration behind the colors, we compute the indexes for the current colors. Each pixel has a value corresponding, of course, to its color.

```
# Indexes for color for each pixel
idx = find_closest_centroids(X, colors)
```

4.1.3 Reconstructing

All the color combination of (rows*cols) number of pixels is now represented by its centroid points. Replace the value of each of the pixels with its centroid point.

Reshape the compressed matrix image to its original dimensions and convert

the raw numbers back to image.

```
# Reconstruct the image
idx = np.array(idx, dtype=np.uint8)
X_reconstructed = np.array(colors[idx, :] * 255, dtype=np.uint8).reshape((w, h, d))
compressed_image = Image.fromarray(X_reconstructed)

# Save reconstructed image to disk
compressed_image.save('dog60.png')
```

4.2 Singular Value Decomposition

4.2.1 Data Pre-Processing

Read the input image from your local disk. The function returns the image as 3 matrices, each corresponding to one channel (R, G and B channels) i.e. segregate the RGB components.

The size of the input image is (rows, cols, 3), flatten all the pixel values to a single dimension of size (rows*cols) and the dimension of each pixel is 3 representing RGB values. The size of the flatten image will be (rows*cols, 3).

```
# open the image and return 3 matrices.
def openImage(imagePath):
    imOrig = Image.open(imagePath)
    im = numpy.array(imOrig)

    aRed = im[:, :, 0]
    aGreen = im[:, :, 1]
    aBlue = im[:, :, 2]

    return [aRed, aGreen, aBlue, imOrig]

# MAIN PROGRAM:
print('*** Image Compression using SVD')
aRed, aGreen, aBlue, originalImage = openImage('dog.jpg')
```

4.2.2 Perform SVD

Compute the SVD decomposition of each color channel of the input image matrix based on the rank. Perform SVD on each color channel i.e. compress the matrix of each color channel separately.

```
# compress the matrix of a single channel
def compressSingleChannel(channelDataMatrix, singularValuesLimit):
    uChannel, sChannel, vhChannel = numpy.linalg.svd(channelDataMatrix)
    aChannelCompressed = numpy.zeros((channelDataMatrix.shape[0], channelDataMatrix.shape[1]))
    k = singularValuesLimit

    leftSide = numpy.matmul(uChannel[:, 0:k], numpy.diag(sChannel)[0:k, 0:k])
    aChannelCompressedInner = numpy.matmul(leftSide, vhChannel[0:k, :])
    aChannelCompressed = aChannelCompressedInner.astype('uint8')
    return aChannelCompressed

singularValuesLimit = 1000

aRedCompressed = compressSingleChannel(aRed, singularValuesLimit)
aGreenCompressed = compressSingleChannel(aGreen, singularValuesLimit)
aBlueCompressed = compressSingleChannel(aBlue, singularValuesLimit)
```

4.2.3 Reconstructing

Form the image of each colour channel from the compressed matrix. Merge the so formed images of each compressed matrix and this forms the final compressed image.

```
imr = Image.fromarray(aRedCompressed, mode=None)
img = Image.fromarray(aGreenCompressed, mode=None)
imb = Image.fromarray(aBlueCompressed, mode=None)

newImage = Image.merge("RGB", (imr, img, imb))

originalImage.show()
newImage.show()
newImage.save("dog1000.jpg")
```

4.3 Comparing efficiencies of K-Means and SVD

Evaluate the compressed image based on Peak Signal to Noise Ratio (PSNR), Structural Similarity Index (SSIM), Compression ratio (CR).

PSNR – It is used as a quality measurement between the original image and a compressed image. Higher the PSNR value, better the image quality.

SSIM – It quantifies image quality degradation caused by image compression. Higher the SSIM value, better the image quality.

CR – It is the ratio between original image size and compressed image size. Higher the compression rate, better the output results.

```
def get_file_size(file_path):
    size = os.path.getsize(file_path)
    return size

def PSNR(original, compressed):
    mse = np.mean((original - compressed) ** 2)
    if(mse == 0):
        return 100
    max_pixel = 255.0
    psnr = 20 * log10(max_pixel / sqrt(mse))
    return psnr
```

```
original = cv2.imread("lena.png")
compressed = cv2.imread("lena5.png",1)

value = PSNR(original, compressed)
value = round(value,2)

s = ssim(original, compressed,multichannel=True)
s=round(s,2)

cr = get_file_size('lena.png')/get_file_size('lena5.png')
cr = round(cr,2)

print(f"Compression Ratio is {cr}")
print(f"PSNR value is {value} dB")
print(f"SSIM value is {s}")
```

```
Compression Ratio is 10.67
PSNR value is 29.39 dB
SSIM value is 0.73
```

CHAPTER 5

WORKING DEMO

5.1 K - Means Clustering

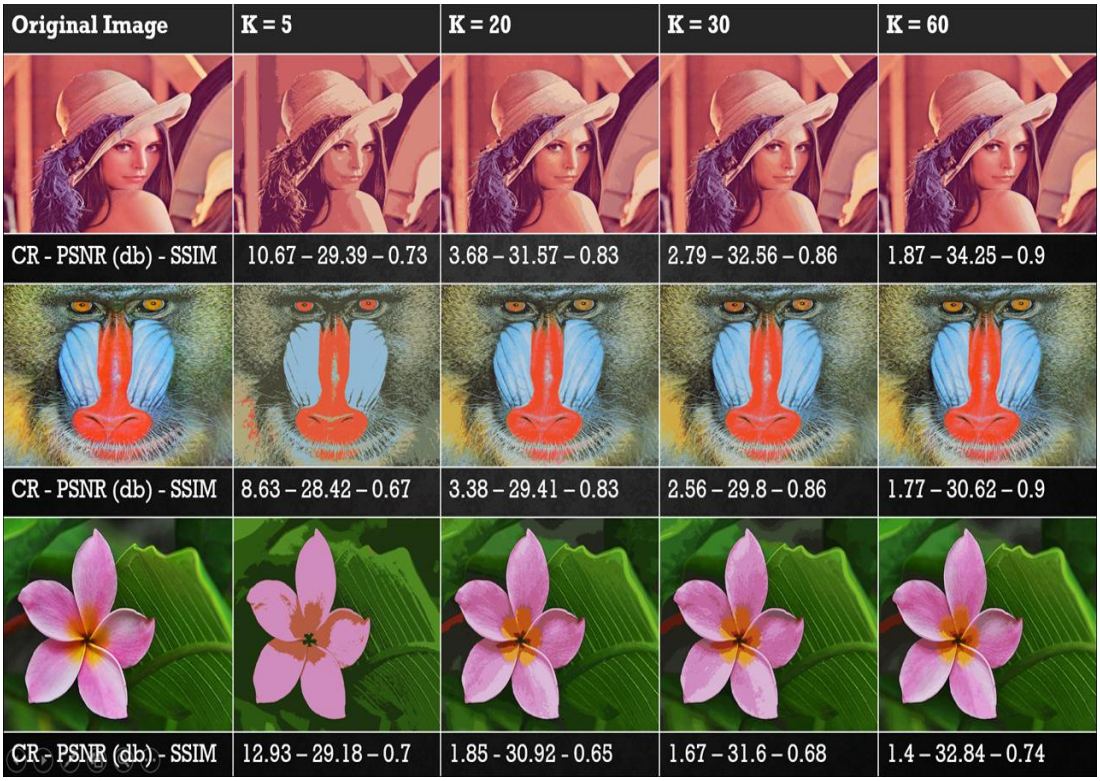
We performed the K-Means Clustering Algorithm on three different images for four different K values and compared them. The results are as follows:

K refers to the number of colors (clusters) in the image.

The smaller value of K indicates a lesser number of colours in the image. So, the quality of image is poor, PSNR and SSIM values are lower and compression ratio is higher.

The larger value of K indicates more number of colours in the image. So, the quality of image increases, PSNR and SSIM values also increase and compression ratio decreases.

K value is directly proportional to PSNR and SSIM values and inversely proportional to Compression ratio.



Original Image	K = 10	K = 60	K = 60
			
CR - PSNR (db) - SSIM	7.05 – 29.05 – 0.74	3.55 – 30.67 – 0.83	2.53 – 31.83 – 0.87
			
CR - PSNR (db) - SSIM	4.01 – 31.47 – 0.88	1.69 – 33.74 – 0.94	1.29 – 35.36 – 0.96
			
CR - PSNR (db) - SSIM	5.34 – 30.5 – 0.84	2.68 – 33.21 – 0.91	1.9 – 35.44 – 0.94

Original Image	K = 10	K = 30	K = 60
			
CR - PSNR (db) - SSIM	2.75 – 31.82 – 0.91	1.54 – 35.05 – 0.96	1.18 – 37.07 – 0.97
			
CR - PSNR (db) - SSIM	10.74 – 29.03 – 0.74	5.34 – 30.6 – 0.8	3.78 – 31.81 – 0.84
			
CR - PSNR (db) - SSIM	7.75 – 29.76 – 0.7	2.83 – 31.33 – 0.82	1.98 – 32.56 – 0.87

5.2 Singular Value Decomposition

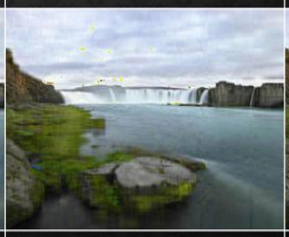
We performed SVD (Singular Value Decomposition) on three different images for four different K values and compared them. The results are as follows:

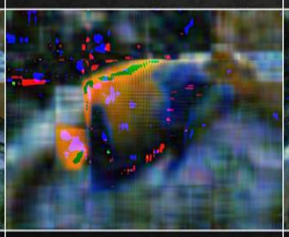
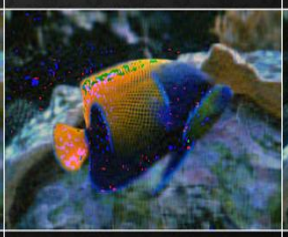
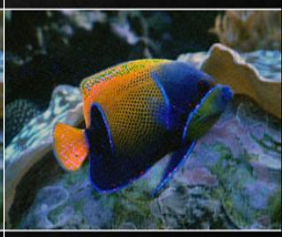
K refers to the number of columns (rank) of the image matrix.

The smaller value of K indicates a lesser number of colors in the image. So, the quality of image is poor, PSNR and SSIM values are lower and compression ratio is higher.

The larger value of K indicates a greater number of colors in the image. So, the quality of image increases, PSNR and SSIM values also increase and compression ratio decreases.

K value is directly proportional to PSNR and SSIM values and inversely proportional to Compression ratio.

Original Image	K = 10	K = 30	K = 60
			
CR - PSNR (db) - SSIM	2.4 – 28.46 – 0.48	1.75 – 29.71 – 0.6	1.62 – 31.05 – 0.7
			
CR - PSNR (db) - SSIM	4.47 – 28.7 – 0.28	2.94 – 28.99 – 0.37	2.27 – 29.33 – 0.5
			
CR - PSNR (db) - SSIM	1.49 – 30.8 – 0.72	1.18 – 32.72 – 0.77	1.01 – 33.8 – 0.83

Original Image	K = 10	K = 30	K = 60
			
CR - PSNR (db) - SSIM	4.47 – 28.99 – 0.37	3.2 – 29.61 – 0.45	2.51 – 29.97 – 0.54
			
CR - PSNR (db) - SSIM	1.95 – 30.68 – 0.69	1.47 – 32.15 – 0.74	1.3 – 33.26 – 0.78
			
CR - PSNR (db) - SSIM	3.05 – 28.95 – 0.47	2.4 – 30.06 – 0.58	2.16 – 31.31 – 0.69

CHAPTER 6

CONCLUSION

6.1 Comparison: K-Means & SVD

CR	Lena				Baboon				Flower			
K	5	20	30	60	5	20	30	60	5	20	30	60
K-Means	10.67	3.68	2.79	1.87	8.63	3.38	2.56	1.77	12.93	1.85	1.67	1.4
SVD	2.37	1.68	1.53	1.38	3.4	2.14	1.87	1.57	2.53	1.85	1.67	1.4

PSNR	Lena				Baboon				Flower			
K	5	20	30	60	5	20	30	60	5	20	30	60
K-Means	29.39	31.57	32.56	34.25	28.42	29.41	29.8	30.62	29.18	30.92	31.6	32.84
SVD	28.88	30.68	31.51	33.26	28.45	28.84	28.93	29.28	29.03	30.92	31.6	32.84

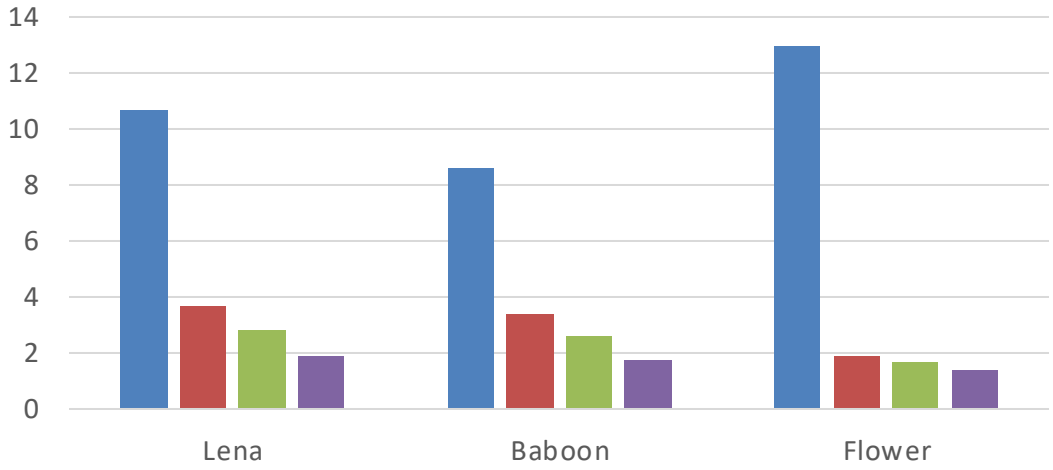
SSIM	Lena				Baboon				Flower			
K	5	20	30	60	5	20	30	60	5	20	30	60
K-Means	0.73	0.83	0.86	0.9	0.67	0.83	0.86	0.9	0.7	0.65	0.68	0.74
SVD	0.59	0.69	0.74	0.84	0.28	0.43	0.51	0.66	0.59	0.65	0.68	0.74

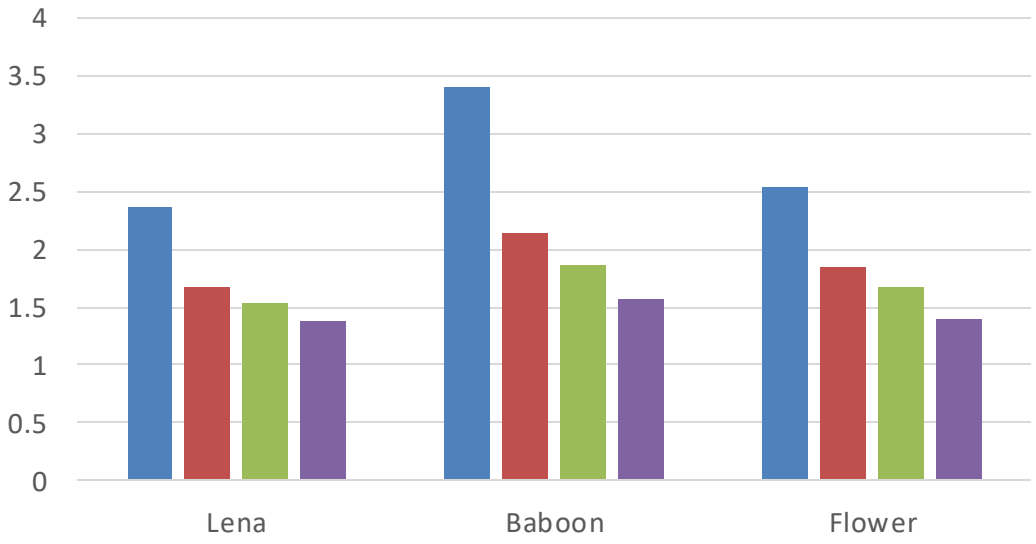
K = 10	CR		PSNR		SSIM	
	K - Means	SVD	K - Means	SVD	K - Means	SVD
Bird	7.05	2.4	29.05	28.46	0.74	0.48
Spider	4.01	4.47	31.07	28.7	0.88	0.28
Waterfall	5.34	1.49	30.5	30.8	0.84	0.72
Lion	2.75	4.47	31.82	28.99	0.99	0.37
Parrot	10.74	1.95	29.03	30.68	0.74	0.69
Fish	7.75	3.05	29.76	28.95	0.7	0.47

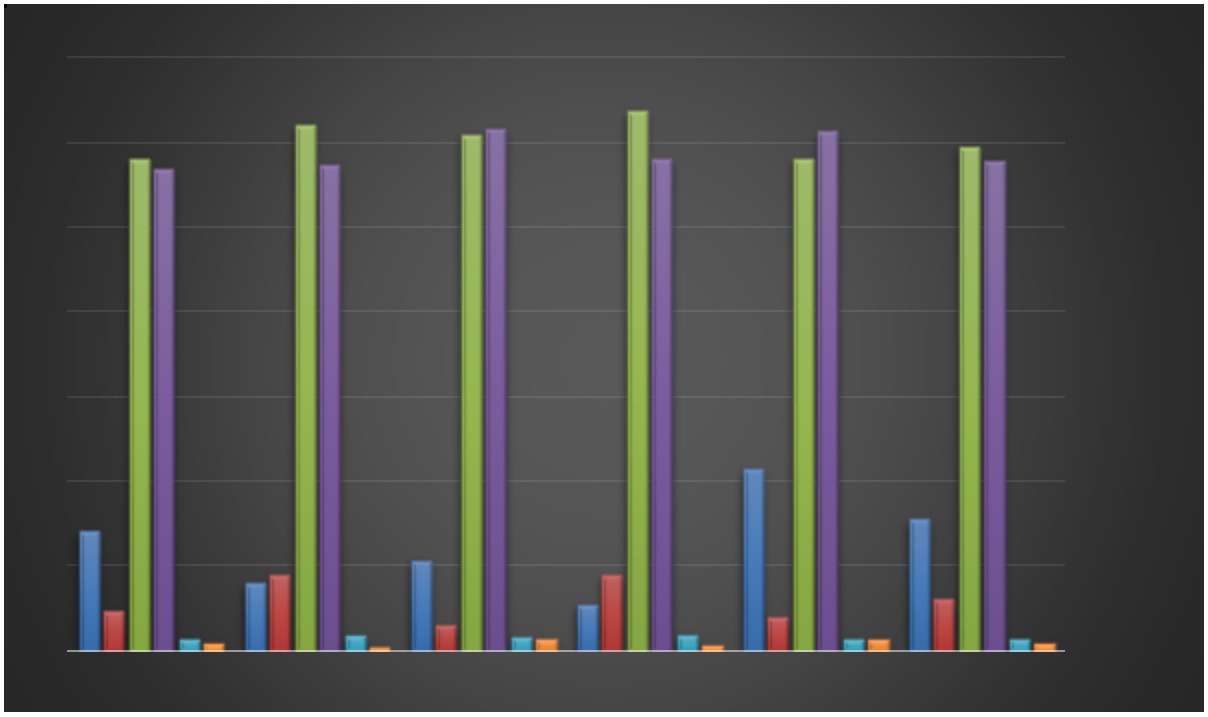
K = 30	CR		PSNR		SSIM	
	K - Means	SVD	K - Means	SVD	K - Means	SVD
Bird	3.55	1.75	30.67	29.71	0.83	0.6
Spider	1.69	2.94	33.74	28.99	0.94	0.37
Waterfall	2.68	1.18	33.21	32.72	0.91	0.77
Lion	1.54	3.2	35.05	29.61	0.96	0.45
Parrot	5.34	1.47	30.6	32.15	0.8	0.74
Fish	2.83	2.4	31.33	30.06	0.82	0.58

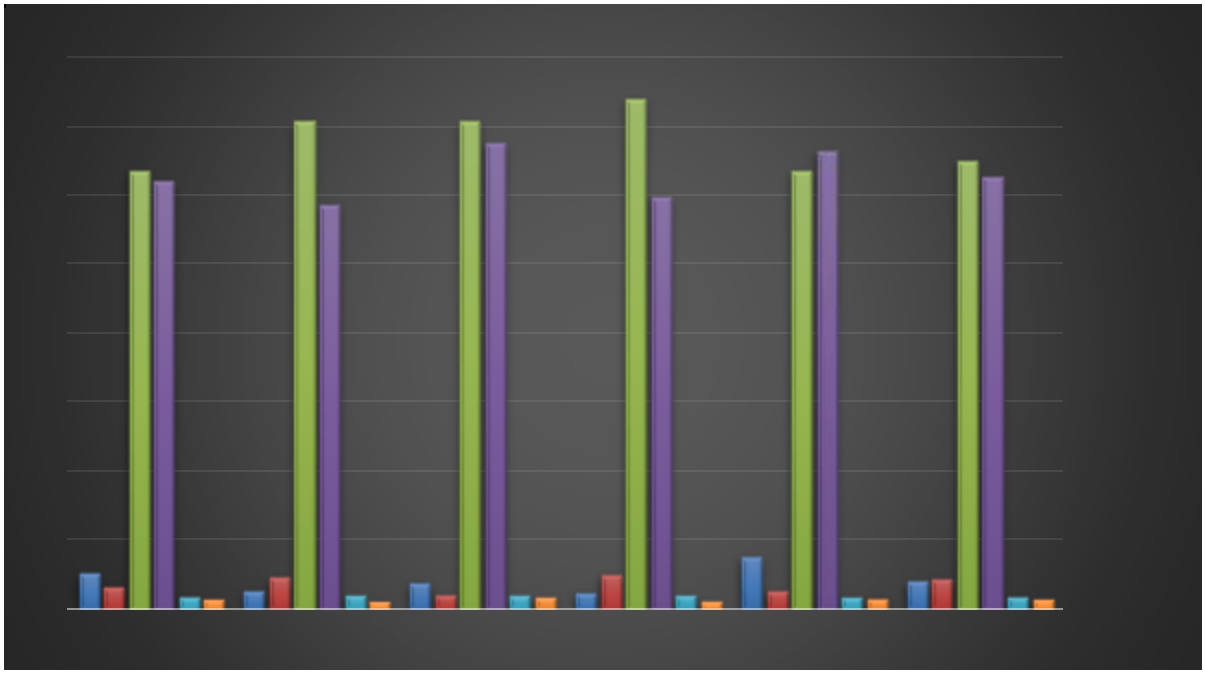
K = 60	CR		PSNR		SSIM	
	K - Means	SVD	K - Means	SVD	K - Means	SVD
Bird	2.53	1.62	31.83	31.05	0.87	0.7
Spider	1.29	2.27	35.36	29.33	0.96	0.5
Waterfall	1.9	1.01	35.44	33.8	0.94	0.83
Lion	1.18	2.51	37.07	29.97	0.97	0.54
Parrot	3.78	1.3	31.81	33.26	0.84	0.78
Fish	1.98	2.16	32.56	31.31	0.87	0.69

6.1 Plot: K-Means vs SVD




4
3.5
3
2.5
2
1.5
1
0.5
0
Lena
Baboon
Flower





From the above comparison tables, we can observe that:
For same K value, K-Means gives better compression ratio in almost all images, better PSNR value and SSIM value in all images. Hence, K – Means is more efficient in compressing the image without degrading the quality of the original image.
But, K – Means takes more time to compress the image as compared to SVD.
These techniques can be used for both personal and business purposes.

REFERENCES

1. IMAGE COMPRESSION METHODS BASED ON TRANSFORM CODING AND FRACTAL CODING - V. Yaswanth Varma, T. Nalini Prasad, N. V. Phani Sai Kumar ECE Department, SRKR Engineering College, India

2. A COMPARATIVE ANALYSIS OF IMAGE COMPRESSION TECHNIQUES: K MEANS CLUSTERING AND SINGULAR VALUE DECOMPOSITION - R. Gomathi and R. Aparna Department of Computer Applications, Madras University, India

3. Analysis And Study Of K-Means Clustering Algorithm - Sudhir Singh and Nasib Singh Gill, Dept of Computer Science & Applications, M. D. University, Rohtak, Haryana

4. Image Compression using Singular Value Decomposition - Miss Samruddhi Kahu, Associate Engineer, Marvell Technologies & Ms. Reena Rahate, Professor, Shri Ramdeobaba College of Engg. and Management Nagpur

5. An Improved SVD based Image Compression - Kapil Mishra, Satish Kumar Singh, and P. Nagabhushan Information Technology Department Indian Institute of Information Technology Allahabad, India