www.ijecs.in
International Journal Of Engineering And Computer Science
Volume 9 Issue 06 June 2020, Page No. 25075-25084
ISSN: 2319-7242 DOI: 10.18535/ijecs/v9i06.4463

Comparitive Analysis and Findings on Dct & Lbg Compression Techniques

[1]Mr. Moayad Al Falahi, [2]Dr. Janaki Sivakumar[*]

[1]Student, Bachelor of Software Engineering, Department of Computing Muscat College
[2]Assistant Professor Department of Computing Muscat College

Abstract:
The main objective of this project is to develop an application to find the best compression technique to store Muscat College students' photographs in less storage. MATLAB software will be used to develop a Graphical User Interface GUI application and implement two image compression techniques which are lossless compression using the DCT algorithm and lossy compression using the LBG algorithm. The application shall allow the user to select and test a sample image by applying both these techniques for any student image he\she selects in order to compare the results by display the image after compression and the histogram to find which the most suitable compression technique is. Also, the application shall show the size of images before and after applying the compression process and show the compression ratio and relative data redundancy of compressed image/images. The main functionality is that the application shall allow the user to do bulk processing to apply image enhancement and image compression technique to enhance and compress all the photographs of students and store them in less space.

1. Introduction

Nowadays, the digital image can be processed by a digital computer, where we are able to edit, change and improve the quality of an image [1]. The first process to get the image in the image processing procedure is called as image acquisition, where we obtain the image using a different type of camera devices such as mobile phones, digital cameras, satellite, etc.

In social media applications such as Facebook, Instagram, Twitter, WhatsApp, etc. there are many images that are uploaded every day. According to the statistic that in 2014 there are 2088 digital images are uploaded every second. That's 1.8 billion digital images every day and 657 billion digital images per year [2]. Also, in many of the organization such as universities, colleges, schools, police, etc. a large number of digital images need to be stored in the database. In the colleges and universities every new year there are new students are joining college and their information needs to be stored in the database including a photograph of each student. Therefore, they will be a large number of digital images that consume a large space from storage. The most important thing for colleges and universities is to store these photographs in less space and to be valid in the future to recognize students' faces. In this research, an application shall be developed to find the best compression technique to store Muscat College students' photographs in less storage.

There are different image compression techniques used in different applications to store a large number of digital images in less storage such as Huffman code, run-length code, Discrete Cosine Transform DCT, Linde Buzo Gray LBG, etc. some type of compression techniques called lossless compression which maintain the quality of the images and some type of compression techniques called lossy compression which will not maintain the quality of the images but will save more storage.

2. Compression Algorithms

A) Discrete Cosine Transform

DCT is a lossless compression technique applies a transformation to the image by converts the spatial domain into the frequency domain. DCT is a real term and orthogonal. DCT is a sinusoidal function used in JPEG compression. The advantage of the DCT is that the most important information of the image is concentrated in just a few coefficients of the DCT [3].

The DCT formula to find the two-dimensional M-by-N matrix A:

$$B_{pq} = \alpha_p \alpha_q \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} A_{mn} \cos \frac{\pi(2m+1)p}{2M} \cos \frac{\pi(2n+1)q}{2N}, \quad \begin{matrix} 0 \le p \le M-1 \\ 0 \le q \le N-1 \end{matrix}$$

$$\alpha_p = \begin{cases} 1/\sqrt{M}, & p = 0 \\ \sqrt{2/M}, & 1 \le p \le M-1 \end{cases} \quad \alpha_q = \begin{cases} 1/\sqrt{N}, & q = 0 \\ \sqrt{2/N}, & 1 \le q \le N-1 \end{cases}$$

Bpq is representing the DCT coefficients of A. The DCT inverse can be obtained by the following formula:

$$A_{mn} = \sum_{p=0}^{M-1} \sum_{q=0}^{N-1} \alpha_p \alpha_q B_{pq} \cos \frac{\pi(2m+1)p}{2M} \cos \frac{\pi(2n+1)q}{2N}, \quad \begin{matrix} 0 \le m \le M-1 \\ 0 \le n \le N-1 \end{matrix}$$

$$\alpha_p = \begin{cases} 1/\sqrt{M}, & p = 0 \\ \sqrt{2/M}, & 1 \le p \le M-1 \end{cases} \quad \alpha_q = \begin{cases} 1/\sqrt{N}, & q = 0 \\ \sqrt{2/N}, & 1 \le q \le N-1 \end{cases}$$

The inverse DCT equation used to write M-by-N matrix A as sum of MN functions and can be obtained by the following formula:

$$\alpha_p \alpha_q \cos \frac{\pi(2m+1)p}{2M} \cos \frac{\pi(2n+1)q}{2N}, \quad \begin{matrix} 0 \le p \le M-1 \\ 0 \le q \le N-1 \end{matrix}$$

The DCT coefficients Bpq are the basis functions. For example, the 64 basis functions of 8-by-8 matrices are represented as follow (Figure 1):

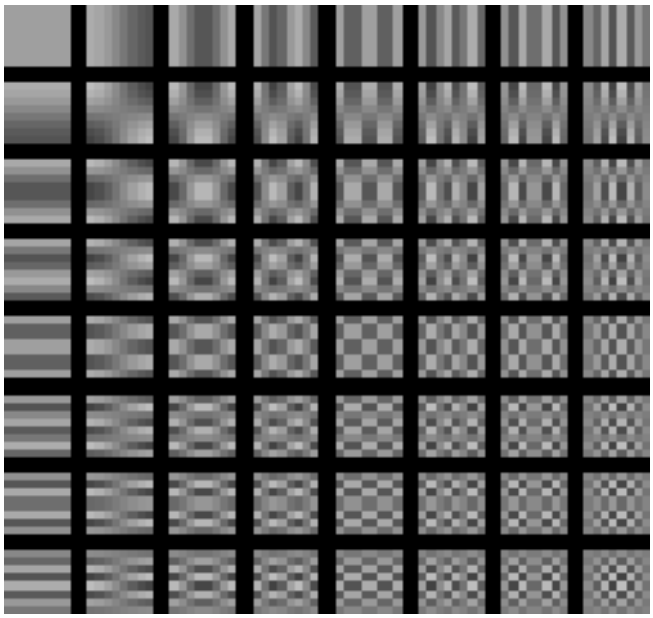


Figure 1: 64 basic functions of 8-by-8 matrix

In the 64 basis functions as shown in Figure 1, the horizontal frequencies values is an increase from top to bottom and from left to right. The frequency value in the upper left position is called a DC basis function.

$$T_{pq} = \begin{cases} \frac{1}{\sqrt{M}} & p = 0, \qquad 0 \le q \le M-1 \\ \sqrt{\frac{2}{M}} \cos\frac{\pi(2q+1)p}{2M} & 1 \le p \le M-1, \quad 0 \le q. \le M-1 \end{cases}$$

There are two functions can be used to calculate DCT in MATLAB. The first function that used in this application is dctmtx, this function is suitable for the 8-by-8 or 16-by-16 transform matrix. The second function is dct2 which is based on the FFT algorithm. The M-by-M transform matrix T is given by:

The one-dimensional image is calculated by T*A which is M-by-M matrix A where T is transform matrix and A is the original image. The two-dimensional image is calculated by T*A*T' where T is transform matrix, A is the original image and T' is cosine transform matrix [4].

The algorithm of DCT works by reading the image and takes the red color value of the image then converts it to a double image matrix. Then Compute the two-dimensional DCT of 8-by-8 blocks in the image using the function dctmtx which returns the N-by-N DCT transform matrix [4]. After that, it will apply DCT on each block and design a filter to remove the coefficients in a zigzag manner. The same procedure shall be applied to the green color value and blue color value and then combine the output of the three-color components in one image using the cat function [7].

B) Linde Buzo Gray

The full form of LBG is Linde Buzo Gray. LBG is a lossy compression technique that proposes a vector quantization algorithm based on a training sequence [5]. LBG is the most popular approach for obtaining a vector quantizer codebook. The codebook is a source input group in some vector to find the closest value with codebook values. The set of quantizer output points is called a codebook of the quantizer. LBG is based on a k-means algorithm that applies a clustering procedure to a large set of output vector called a training set. K-means set an initial set of k representative patterns. This representative pattern is updated by computing the centroid of training set vectors. In LBG the distortion from one iteration to the next will not increase [6].

The following block diagram (Figure 2) describes how the vector quantization works:

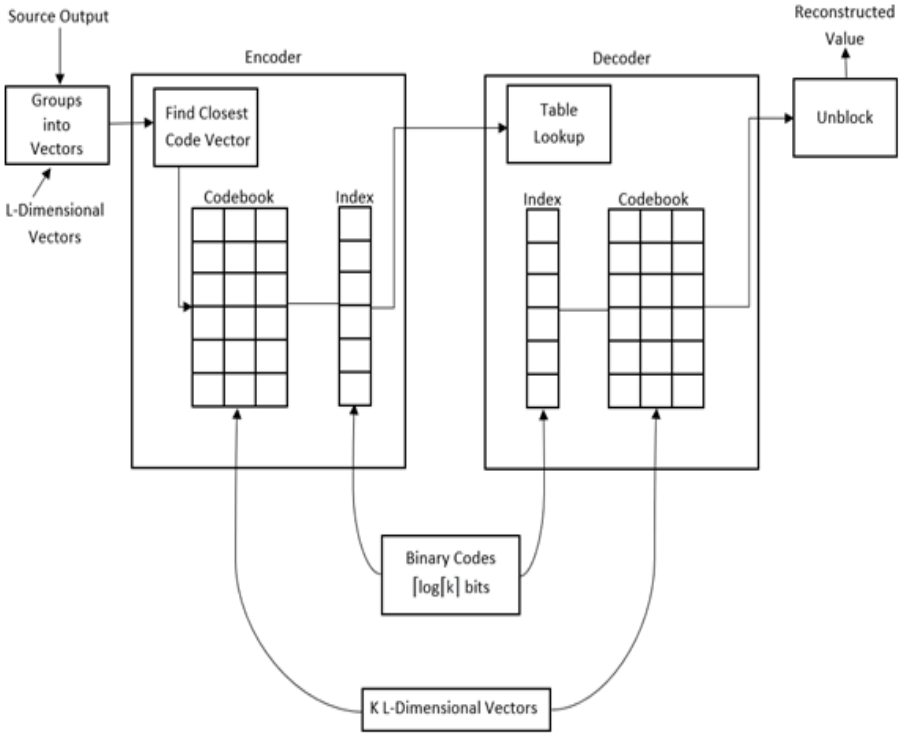


Figure 2: block diagram

In the block diagram as shown in Figure 2, there is source output which is groups into vectors which is into L-Dimensional vectors. Also, there are encoder and decoder sides. In the encoder side, there is a block which is find closest code vectors. In both encoder and decoder sides there is a codebook of L-Dimensional and index of binary code that in form of ceiling of log of ceiling of k bits. In the decoder side, the data are unblocked to obtain the reconstructed value.

How it's work

First, the source output is grouped into blocks (vectors). This vector forms as an input. Then the groups of samples (vectors) are treated as L-Dimensional vectors. The vectors which are present in the codebook are known as code vectors. This code vectors selected as a representation of the vectors generated from the source output which means that the source output is clubbed as a form of vectors. These group of vectors acts as input to the encoder side to find closest match of those vectors which are sent to the decoder side from the codebook of encoder side. Each code vector is assigned a binary index. The input vector is compared with each code vector in order to find the closest value to the input vector and that closest value has a corresponding index value which is stored in the index. Then that index value will send to the decoder side.

To inform the decoder which code vectors are found, the closest input vectors and the binary code is transmitted or stored to the decoder side. The decoder will have the same code vectors and index value which is in the encoder side. Then the decoder retrieves the index value which is sent from encoder, and lookup that index value from its database and retrieves the corresponding code vector which is having that index value. Then that code vector will send to unblock data to reconstruct the value which is the same as the source output value sent by encoder. There will be no loss of information due to extracting the same output value which is sent by the encoder [15].

3. Test Data Set

4. Results and Discussion

When running the application, the program will keep on asking the user to enter his/her choice initially and after the finish from each process until the user chooses the exit option to close the application as shown in Figure 3.

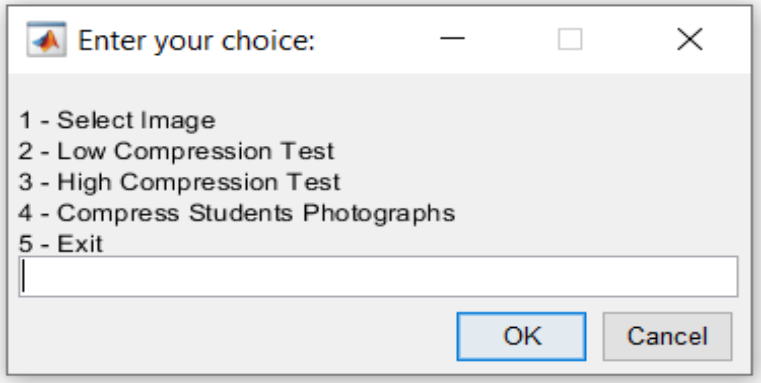


Figure 3:

The program shall display error messages as shown in Figure 4 whenever the user enters incorrect choice.

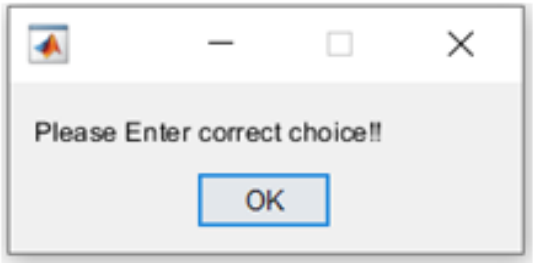


Figure 4:

Users can choose to select a sample image as shown in Figure 5 in order to apply low compression (Lossless) and high compression (Lossy) process to the selected image.

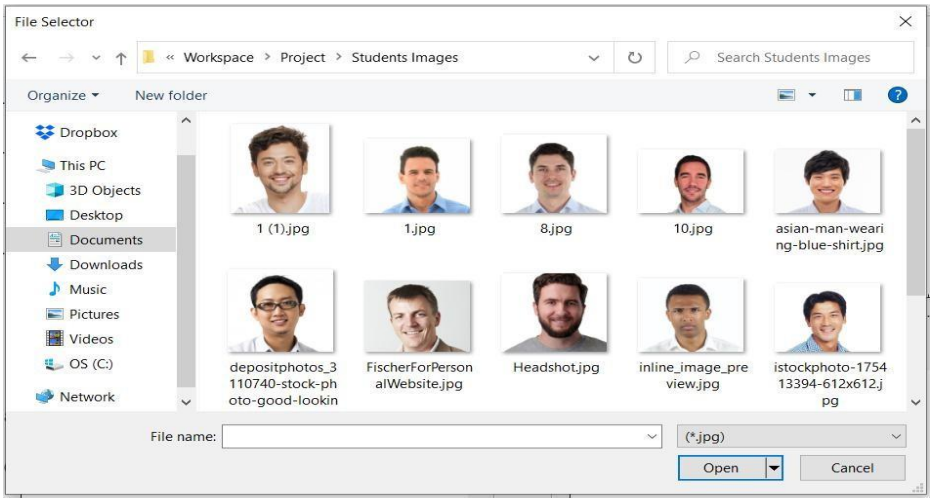


Figure 5:

After selecting an image and applying both compression methods, the program will display the compressed images along with the size of images, compression ratio, relative data redundancy and histogram as shown in Figure 6. Now the user can compare the results and decide the best compression type to be applied for the students' photographs.

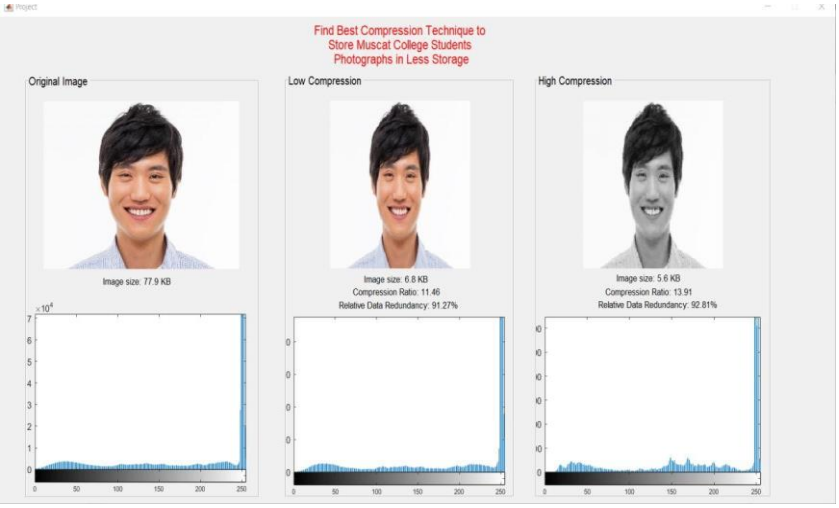


Figure 6:

After decide which is the best compression method and choose the fourth option (Compress Students Photographs), the program shall ask to select the type of images to be compressed and then choose the compression type as shown in Figure 7 and Figure 8 in order to do bulk processing to enhance and compress all photographs of the students.

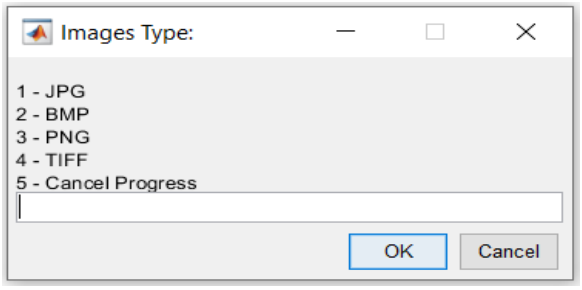


Figure 7:

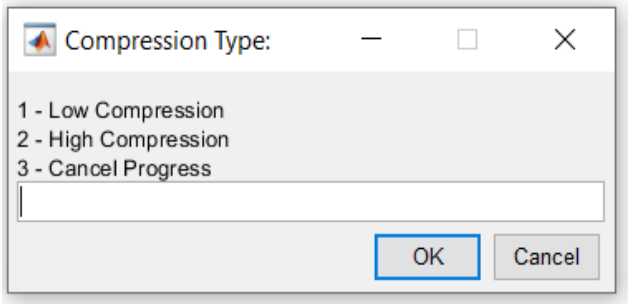


Figure 8:

When user selects compression type, the program will compress and enhance all the images which are stored in a folder called "Students Images" Figure 9 and then store the compressed images in a folder called "Compressed Images". Then convert the compressed images into black and white images and store them in a folder called "Compressed Black and White Images".

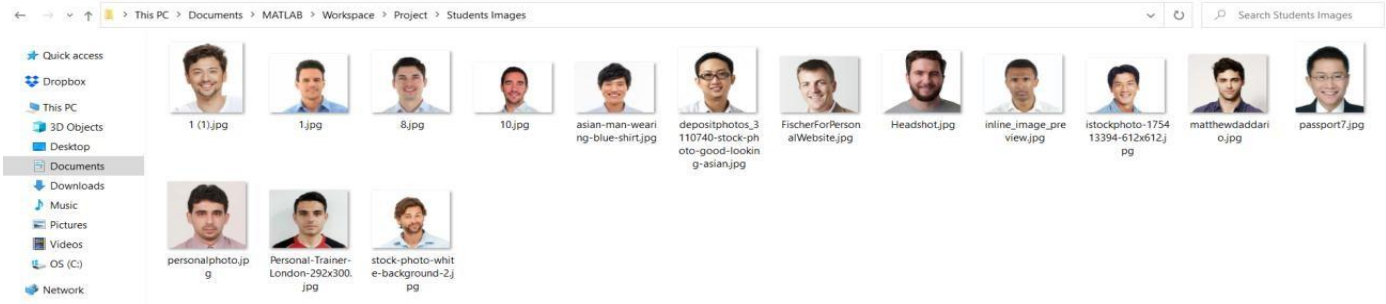


Figure 9:

All compression progress will be shown in the command window as shown in Figure 10 and Figure 11 to provide information to the user about which image is in compression progress in the current time and how many images are compressed. Also, the program will display a success message after finishing the progress and show the size of the images before and after compression along with compression ratio and relative data redundancy.

```
Command Window
Image 1 is in progress...
Image 2 is in progress...
Image 3 is in progress...
Image 4 is in progress...
Image 5 is in progress...
Image 6 is in progress...
Image 7 is in progress...
Image 8 is in progress...
Image 9 is in progress...
Image 10 is in progress...
Image 11 is in progress...
Image 12 is in progress...
Image 13 is in progress...
Image 14 is in progress...
Image 15 is in progress...
Images has been compressed successfuly.

Total size of the original images is:  1377.5 KB
Total size of the compressed images is:  106.9 KB
Compression Ratio: 12.89
Relative Data Redundancy: 92.24%
```

Figure 10: Low Compression Results

```
Image 1 is in progress...
Image 2 is in progress...
Image 3 is in progress...
Image 4 is in progress...
Image 5 is in progress...
Image 6 is in progress...
Image 7 is in progress...
Image 8 is in progress...
Image 9 is in progress...
Image 10 is in progress...
Image 11 is in progress...
Image 12 is in progress...
Image 13 is in progress...
Image 14 is in progress...
Image 15 is in progress...
Images has been compressed successfuly.

Total size of the original images is:  1377.5 KB
Total size of the compressed images is:  82.8 KB
Compression Ratio: 16.64
Relative Data Redundancy: 93.99%
```

Figure 11: High Compression Results

A) Compression Techniques (Results & Comparison)

Figure 12 shows the output of the compressed image using a low compression method (Lossless). The images are compressed using the DCT algorithm, enhanced using the average filter and resized to 264 x 264 x 3.

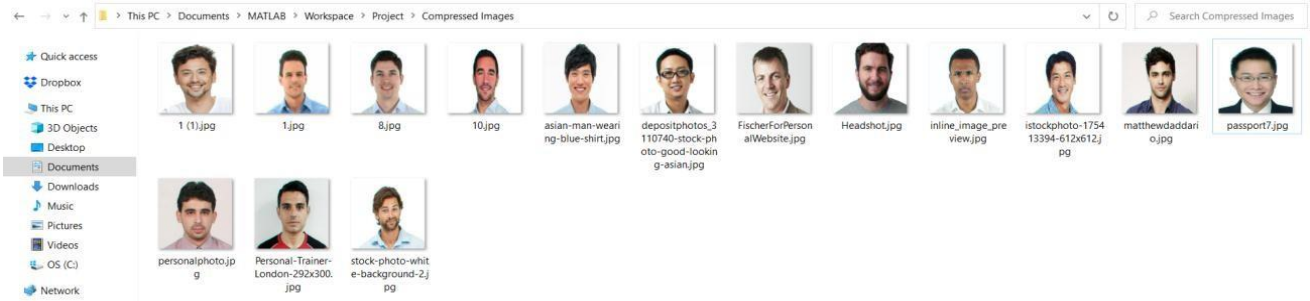


Figure 12:

Figure 13 shows the output of the compressed image using a high compression method (Lossy). The images are compressed using the LBG algorithm, enhanced using the average filter, logarithmic, power law transformation and resized to 264 x 264.

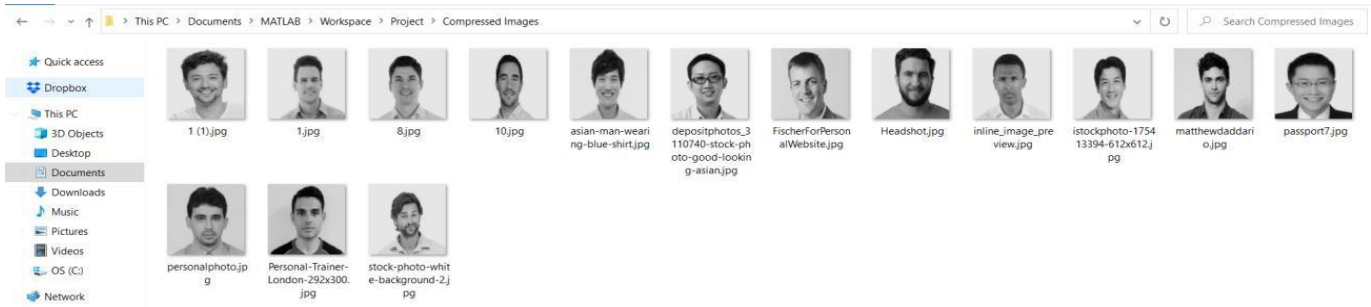


Figure 13:

Figure 14 shows the output of the compressed black and white image after applying a low or high compression method. The compressed images are converted into black and white.

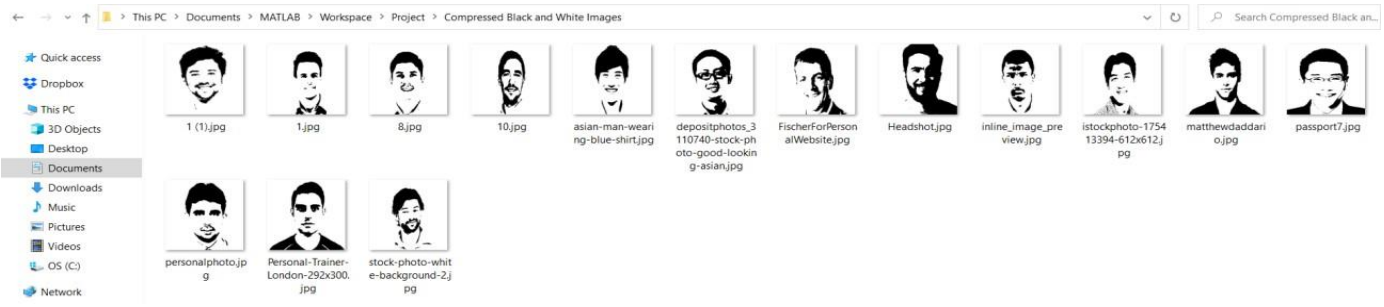
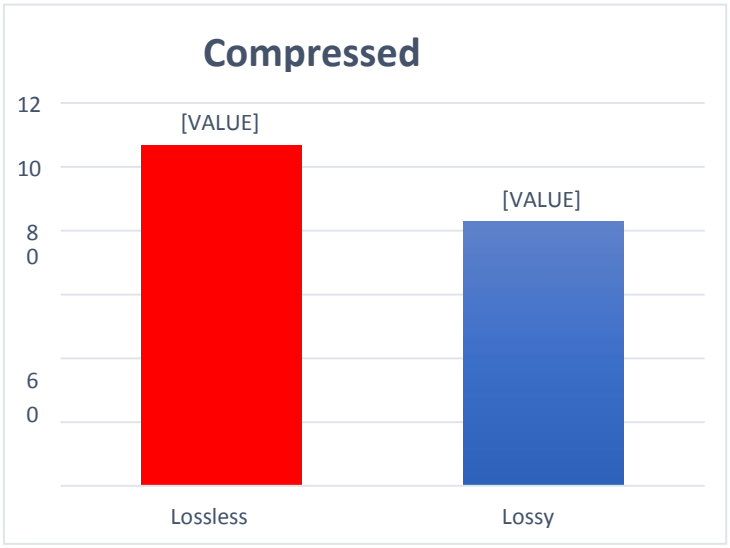


Figure 14:

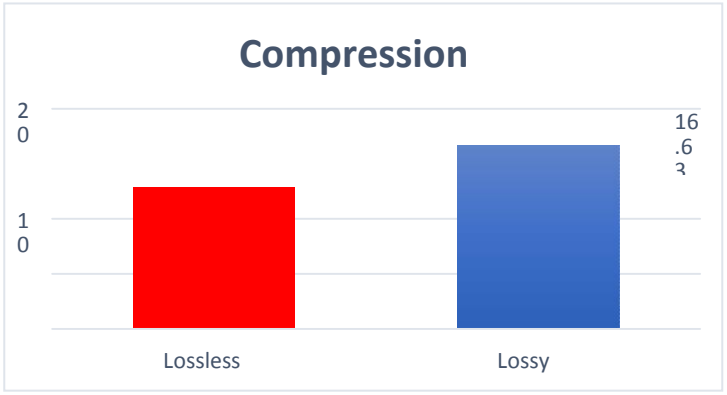
In comparison, the lossless compression algorithm (DCT) has saved the color, quality, and information of the images after compression as shown in Figure 10. On the other hand, the lossy compression algorithm (LBG) has achieved a higher compression ratio and a higher percentage of relative data redundancy and stored the images in less storage as compare to lossless compression.

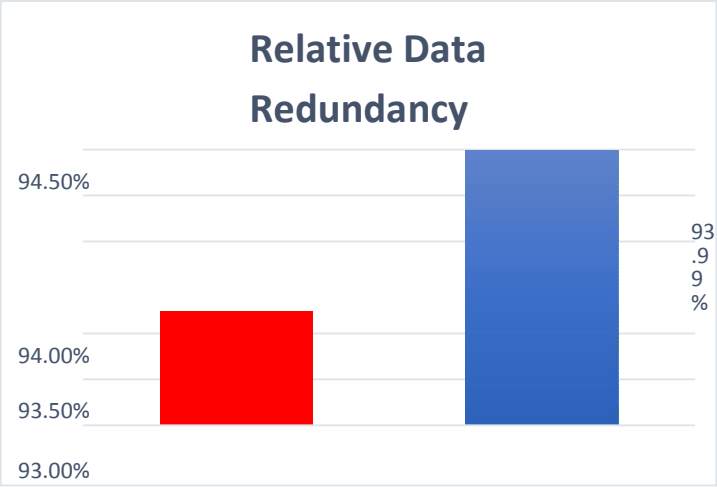


The original images size is 1377.5 KB. The following chart show the images size after compression:

The following chart shows the compression ratio:

The following chart shows the relative data redundancy:





Lossy compression has maintained the information of the images as shown in Figure 13. Although Lossy compression did not save the color like lossless compression. However, Lossy compression is the most appropriate compression technique to store the Muscat College students' photographs in less storage, because the main goal as given in the scenario in the introduction is to store the images in less storage and still faces can be distinguished in the future, but the color and the quality is not much important.

5. Conclusion

All in all, after developing the GUI application to find the best compression technique to store the students' photographs in less storage, it can be concluded that:

- In lossless compression, only the two-dimensional matrix M x N is affected but in lossy compression, the three-dimensional matrix M x N x 3 is affected.
- Lossless compression using the DCT algorithm is suitable to compress the image without quality loss.
- The compression techniques are very important to reduce utilized space in memory in order to store more information.
- Lossy compression using LBG is not suitable to save the color of the images. But is suitable to save the images in less space as compared to lossless compression.
- Lossy compression using LBG is suitable for high-resolution images only, therefore resizing the low-resolution image to the high resolution before compression is most, otherwise, the quality of the image will be lost.
- Lossy compression is good to achieve a high compression ratio and a high percentage of relative data redundancy.

6. References

[1.] Sachin Seth, "What are the advantages of digital image processing", www.quora.com, Mar. 22, 2017. [online]. Available: https://www.quora.com/What-are-the-advantages-of-digital- image-processing. [Accessed Oct. 19, 2019].

[2.] Mary Meeker, "Internet Trends 2018", www.kleinerperkin.com, May. 30, 2018. [online]. Available: https://www.kleinerperkins.com/files/INTERNET_TRENDS_REPORT_2018.pdf. [Accessed January. 2, 2020].

[3.] Shrenik Jain, "(DCT) Discrete Cosine Transform in image processing", www.youtube.com, December. 13, 2017. [online]. Available: https://youtu.be/tW3Hc0Wrgl0. [Accessed January. 3, 2020].

[4.] MathWorks, "Discrete Cosine Transform", www.mathworks.com. [online]. Available: https://www.mathworks.com/help/images/discrete-cosine-transform.html. [Accessed January. 3, 2020].

[5.] MathWorks, “IMAGE COMPRESSION USING LBG ALGORITHM”, www.mathworks.com. [online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/10457-image-compression-using-lbg- algorithm. [Accessed January. 3, 2020].
[6.] ITechnica, “Linde Buzo Gray (LBG) algorithm step by step with example”, www.youtube.com, May. 1, 2018. [online]. Available: https://youtu.be/_S-5M8Zf6hc. [Accessed January. 3, 2020].
[7.] Dr.K.Somasundaram & Dr.T.Kalaiselvi, “Image Compression using Discrete Cosine Transform(DCT) using MATLAB blockproc() Function”, www.researchgate.net, February. 23, 2015. [online]. Available: https://www.researchgate.net/publication/272676573_Image_Compression_using_Discrete_Cosine_TransformDCT_using_MATLAB_blockproc_Function. [Accessed January. 3, 2020].
[8.] MathWorks, “How can I display image size, dimensions and color type in the GUI?”, www.mathworks.com, Aug. 18, 2014. [online]. Available: https://www.mathworks.com/matlabcentral/answers/151331-how-can-i-display-image-size- dimensions-and-color-type-in-the-gui. [Accessed December. 12, 2019].
[9.] MathWorks, “msgbox”, www.mathworks.com. [online]. Available: https://www.mathworks.com/help/matlab/ref/msgbox.html. [Accessed December. 12, 2019].
[10.] MathWorks, “imresize”, www.mathworks.com. [online]. Available: https://www.mathworks.com/help/images/ref/imresize.html. [Accessed December. 12, 2019].
[11.] Ulmer Louis, “Image-Compression-Using-Vector-Quantization-with-LBG-algorithm”, getlab.insa-rouen.fr. [online]. Available: https://gitlab.insa-rouen.fr/lulmer/ASI4_LouisULMER/tree/2f2582aeb2d10a136c3259e5e26513416005b28e/TI/projet_quantification/partie_vectorielle. [Accessed January. 2, 2020].
[12.] MathWorks, “How to round a result to two decimal places”, www.mathworks.com, Oct. 24, 2016. [online]. Available: https://www.mathworks.com/matlabcentral/answers/308927-how-to- round-a-result-to-two-decimal-places. [Accessed January. 3, 2020].
[13.] MathWorks, “How do I print a "%" when using fprintf?”, www.mathworks.com, Mars. 26, 2019. [online]. Available: https://www.mathworks.com/matlabcentral/answers/452606-how-do- i-print-a-when-using-fprintf. [Accessed January. 3, 2020].
[14.] MathWorks, “im2bw”, www.mathworks.com. [Online]. Available: https://www.mathworks.com/help/images/ref/im2bw.html. [Accessed January. 5, 2020].
[15.] ITechnica, “Vector Quantization”, www.youtube.com, April. 26, 2018. [online]. Available: https://www.youtube.com/watch?v=MJ1aYSvgAfY. [Accessed April. 2, 2020].