© 2018 IJSRST | Volume 4 | Issue 6 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X

National Conference on Smart Computation and Technology in Conjunction with The Smart City Convergence 2018

Image Compression and Lightweight Android Apps

Rajat Jhakal[*1], Rishika Nagar[2], Rashi Jain[3]

[1]Computer Engineering, Poornima Institute of Engineering & Technology, Jaipur, Rajasthan, India

[2] Computer Engineering, Poornima Institute of Engineering & Technology, Jaipur, Rajasthan, India

[3] Computer Engineering, Poornima Institute of Engineering & Technology, Jaipur, Rajasthan, India

ABSTRACT

In Android operating system, to achieve greater performance efficient memory allocation and consumption is required. It is very important to efficiently use and manage the internal and external memory space present inside the mobile operating system. Various techniques have been used and implemented to reduce the memory usage in android. One of the techniques for better utilization of memory is image compression in android. The objective of image compression is to reduce the redundancy of the image and to store or transmit data in an efficient form. Based on this compression will be done on image by keeping its quality intact and the possibility of data loss will be minimal.

Keywords: Image Scaling, Android OS, Application, Memory Consumption, Color Quantization, Predictive coding, Image Compression

I. INTRODUCTION

Nowadays we are facing the increasing use of images in many parts of our life. Smart phones, 3D vision systems, satellites, cameras, medical equipment etc. All of these uses or generate image for different tasks. For example, the images, for competitive examination or profile pictures we need to upload the compressed image. We need to save or transmit these images on our devices, then because of the limitation in disk space and channel bandwidth we almost always need image compression for decreasing the size of data, which must be save or transmit. There are some methods/techniques for image compression based on several criteria and conditions. [1]

Some of these criteria/conditions are compression time, compression ratio, quality of compressed image etc. Image compression is reducing the size of a graphics file in bytes without distorting the quality of the image. [2] The reduction in file size allows more images to be stored in a given amount of disk or memory space as well as reduces the time required for images to transmit over the Internet or downloaded from android. The JPEG method is more often used for photographs, while the PNG method is designed to work for online viewing applications like web browsers so it is fully stream able with a progressive display option. [3] The PNG file format was introduced as a free, open-source successor to GIF. Android SDK multiple drawable directories exist for different screen resolutions. There are low, medium, and high DPI specific directories such as drawable-ldpi, drawable mdpi and drawable-hdpi respectively. This allows you to create images at different DPI to enhance the appearance of your application. [7]

Figure 1. Basic Idea of Compression

II. IMAGE COMPRESSION TECHNIQUE

Image compression is very vast area where new concepts and
methods are used day-by-day. Compression is the art of representing the information in a compact form rather than its original or uncompressed form. In other words, size of a file can be reduced using compression methods. [8] Image compression addresses the problem of reducing the amount of data required to represent a digital image. It is a process intended to yield a compact representation of an image, thereby reducing the image storage/transmission requirements. [6]

A. ADVANTAGES OF IMAGE COMPRESSION

1. It provides a potential cost savings associated with sending less data over mobile phone network.
2. It not only reduces overall execution time but also storage requirements allowing greater performance.
3. It also reduces the probability of transmission errors since fewer bits are transferred.
4. It also provides a level of security against illicit monitoring.
5. It is also important efficiently use and manage the internal and external memory requirements of mobile OS. [11]

B. The image compression techniques are broadly classified into two categories:

1. Lossless Technique
2. Lossy Technique

III. PROPOSED METHOD

A. Color Quantization

By definition color quantization is a process that reduces the number of distinct colors used in an image, usually with the idea/goal that the new obtained image will be as optically similar as possible to the original image. Color quantization is a process of segmenting a color space of an image into color regions. Each region can then be represented by a representative color, generally the process is used to represent a color image using a bunch of representative colors which take fewer bits to represent the image. However, this representation scheme introduces image distortion that need to be minimized. The distortion caused can be calculated as the total quantization error which is actually the sum of squared distances between actual pixel colors and their color representatives. The distance between the color point is measured using Euclidian Distance.

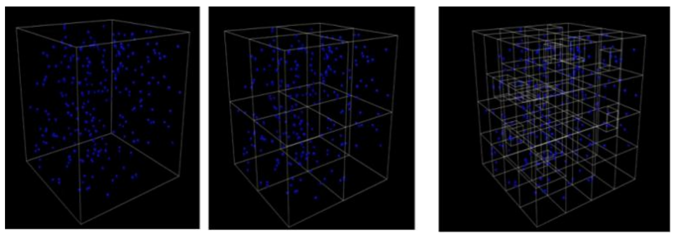


Figure 2. OcTree Representation

The principal methods of implementing color quantization are Octree representation and Euclidian Distance.

Most standard techniques sometimes consider color quantization as a method of clustering points in 3-D space, where the points represent colors found in the original image and the three axes represent the three-color channels. Almost any 3-D clustering algorithm can be applied to color quantization, and vice versa. After the clusters are located, typically the points in each cluster are averaged to obtain the representative color that all the colors in that cluster are mapped to. The 3-color channels are usually red, green and blue.

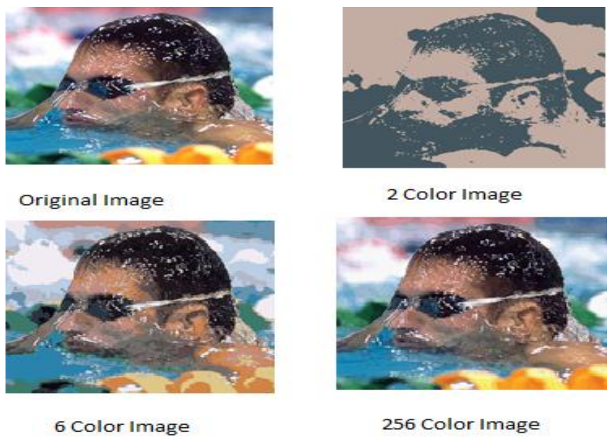


Fig 3: Color Quantized Image 1

B. Advantages of Color Quantization Method

1. Enables efficient compression of certain types of images.
2. Critical for displaying images due to memory limitations.
3. This representation scheme introduces image distortion that need to be minimized.

C. Disadvantages of Color Quantization Method

1. May cause distortion where high definition image is required.
2. Distortion can be minimized by Euclidian Distance.

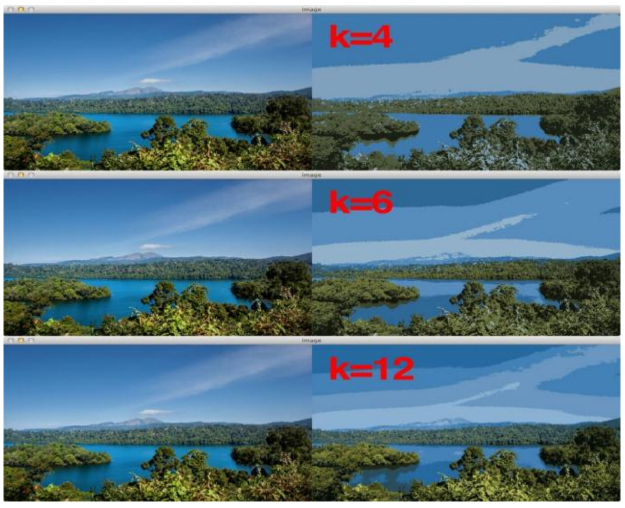


Figure 4. Color Quantized Image 2

D. Predictive Coding

In Predictive coding, it will calculate the predicate color value of each pixel based on the quantized color of adjacent pixel. It is an approach that achieves good compression without significant overload. This coding scheme will take original image and stores all the RGB values of images into a Matrix, then color quantization is performed on the matrix where the image is divided into number of regions and RGB color values are identified. This RGB color values are stored in matrix format called color histogram matrix according to their RGB color axis of intensity values. From this histogram matrix, a centroid for each region is computed. After Color quantization, predictive coding is used to find the predicate value for each color, this predicate value is calculated according to the location of the current pixel's value.

For most of the images, there are redundancies among the adjacent pixel values of the image matrix; i.e., adjacent pixels are highly correlated. Thus, a current pixel can be predicted reasonably well based on a neighborhood of pixels. The prediction error, which is calculated by subtracting the prediction value from the original pixel, has a smaller entropy than the original pixels. Hence, the prediction error can be encoded with fewer bits. In the predictive coding, the correlation between adjacent pixels is removed and the remaining values are encoded.

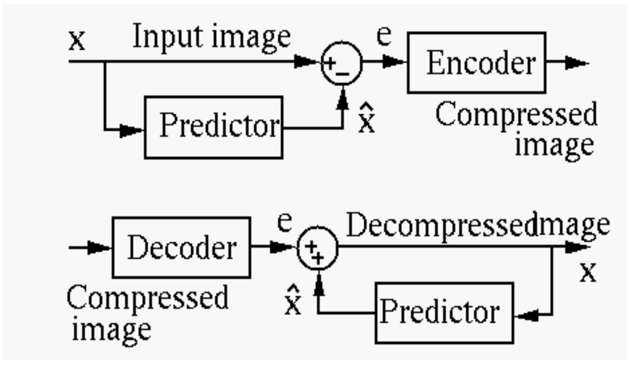


Figure 5. Predictive Coding Block Diagram

Predictive coding is based on eliminating the inter pixel redundancies closely spaced pixels by extracting and code only the new information in each pixel where the new information explains the difference between actual and predicted pixel values.

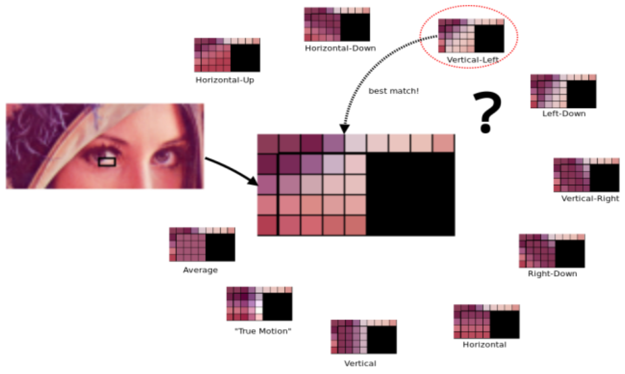


Figure 6. Predictive Coding Pictorial Representation

E. Advantages of Predictive Coding Method

1. Effective method and makes use of histogram and quantization.
2. Easy to implement method.

F. Disadvantages of Predictive Coding Method

1. Drawback is probability of loosy coding and compression.

IV. PROPOSED METHOD IN LIGHTWEIGHT ANDROIOD APPLICATIONS.

In lightweight android apps, the idea of compression is to resize the image and scaling of image is needed. Image scaling is the process of resizing a digital image. Scaling is a significant process that involves a trade-off between efficiency, smoothness and sharpness. In android for images scaling it will consider three factors height of the image, width of the image, resolution of image measured in a DPI. DPI simply means the "Dots Per Inch" in the image. That simply means it's the measure of the resolution of your image based on the number of pixels or dots per inch.

A. Device Oriented Scaling

In device, oriented scaling, it depends upon the different screen sizes and densities of the individual devices. The system performs scaling and resizing to make your application work on different screens, it fit the screen on their devices

B. User Customize scaling

In this type of scaling it depends upon the user what is the scaling factor to compress the image. We can scale the image to reduce the size of images as per the requirement of the user. But, it will reduce the quality of the image. We need to decide the scaling factor, it will not reduce the quality of the image.

C. System Architecture in Context of Image Compression

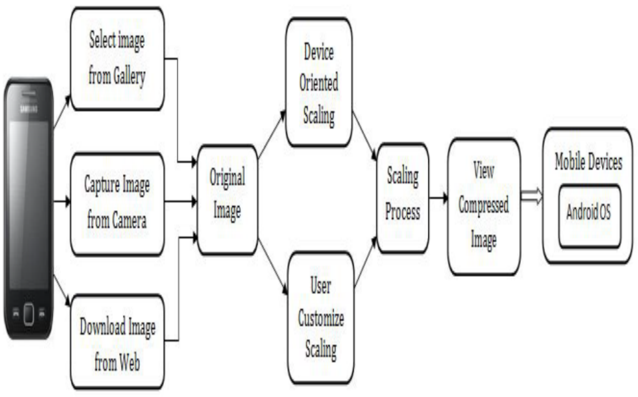
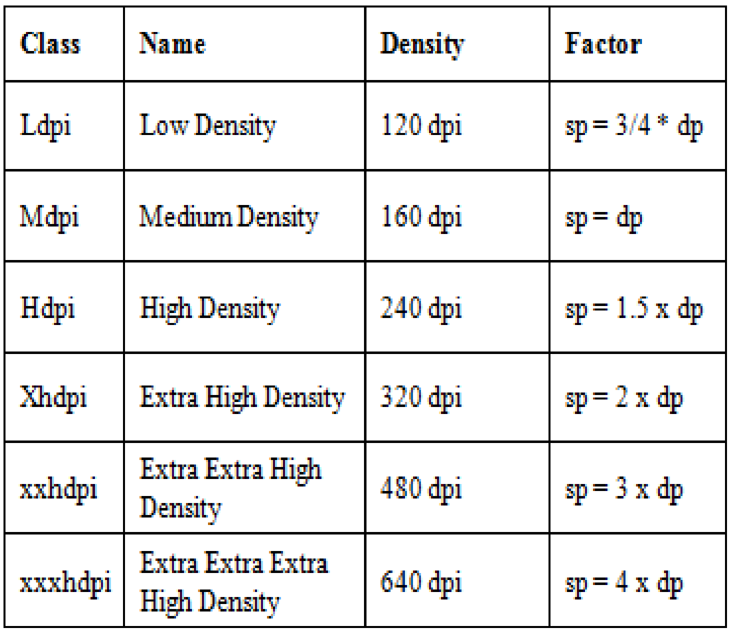


Figure 7. Proposed System Architecture

D. Figurative View of Compression in Android Apps



Class	Name	Density	Factor
Ldpi	Low Density	120 dpi	sp = 3/4 * dp
Mdpi	Medium Density	160 dpi	sp = dp
Hdpi	High Density	240 dpi	sp = 1.5 x dp
Xhdpi	Extra High Density	320 dpi	sp = 2 x dp
xxhdpi	Extra Extra High Density	480 dpi	sp = 3 x dp
xxxhdpi	Extra Extra Extra High Density	640 dpi	sp = 4 x dp

Figure 8. DPI Classes in Android

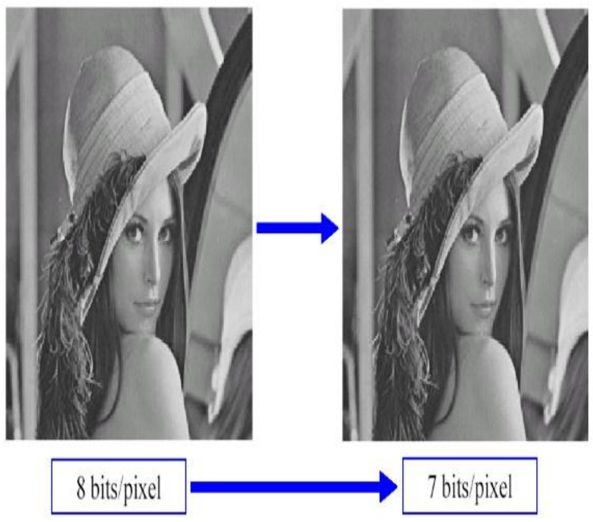


Figure 9. Compressed image in Android

E. Functional Mapping of Original Image and Compressed Image

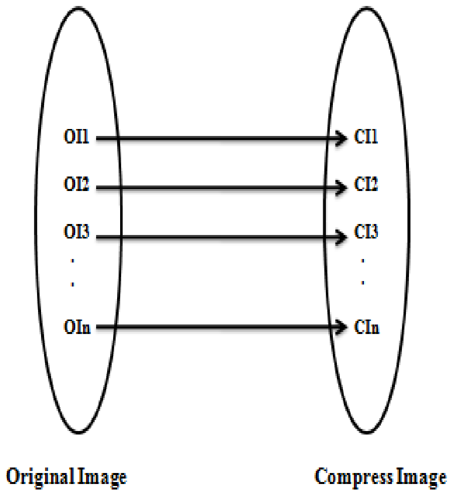


Figure 10. Functional Mapping in Android

IV. CHALLENGES

Image compression is a highly-recommended technique only when you have some quality assessment tools to check for the results as it is a subjective process the error detection and quality assessment becomes a need to verify our works. The main challenge that it faces is the losing of data - sometimes to achieve compression we may lose data so it becomes necessary to know that hoe much compression is needed and how much is beneficial to our project.

V. BENEFITS

Memory allocation and power consumption goes hand in hand and requires to be optimized in this high-tech era therefore the need of lightweight apps arises. Because of these kinds of apps peaks in the areas of memory and power has seen significant changes. Some benefits of the project work can be seen in following areas.

a. Battery life.
b. Transmission time and data consumption.
c. Memory requirements.
d. Performance issues.

VI. FUTURE WORK

The future of lightweight android apps has a rise in tide kind of scenario it may establish base for future improvement techniques in memory requirements and will fulfill needs for optimization processes.

VII. CONCLUSION

Image processing has wide verity of applications leaving option to the researcher to choose one of the areas of his interest. Lots of research findings are published but lots of research areas are still untouched. Moreover, with the fast computers and signal processors available in the 2000s, digital image processing has become the most common form of image processing and generally, is used because it is not only the most versatile method, but also the cheapest.

VIII. REFERENCES

[1] Robert D. Dony, Simon Haykin, Neural Network Approaches to Image Compression, Proceedings of the IEEE, vol. 83, no. 2, February 1995.

[2] http://developer.android.com/guide/practices/screens_support.html

[3] http://www.ccbirding.com/dcsig/howto/dpi/dpi.pdf

[4] Ming-Bo Lin, Jang-Feng Lee, and Gene Eu Jan, “A Lossless Data Compression and Decompression Algorithm and Its Hardware Architecture,” IEEE Trans. Very Large Scale Integer. (VLSI) Syst., vol. 14, no. 9, pp 925-936, September 2006.

[5] Mamta Sharma, " Compression Using Huffman Coding", IJCSNS International Journal of Computer Science and Network Security, Vol.10 No.5, May 2010.

[6] http://www.higherpass.com/Android/Tutorials/Working-With-Images-In-Android/3/

[7] http://developer.sonymobile.com/2011/0627/how-to-scale-images-for-your-ndroid-application/

[8] http://en.wikipedia.org/wiki/Image_file_formats

[9] http://stefan222devel.blogspot.in/2012/10/android-screen-densities-sizes.html

[10] R. N. Neelamani, R. Queiroz, Z. Fan, and R. Baraniuk (2006), JPEG Compression History Estimation for Color Images, IEEE Transactions on image processing, Vol. 15 No. 6, pp. 1365-1378

[11] Y. Sirisathikul, S. Auwatanamongkol, and B. Uyyanonvara (2004), Fast Color Image Quantization Using Distance between Adjacent Colors along the Color Axis with the Highest Color Variance, Pattern Recognition Letter, Vol. 25, No. 9, pp. 1025-1043.