International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161

Available at http://inpressco.com/category/ijcet

Research Article

Image Compression Algorithm using Predictive Coding based on Color Quantization for Android Mobile Devices

Rupali Sachin Vairagade[Á*] and Rekha Kulkarni[Á]

[Á]Pune Institute of Computer Technology, Department of Computer Engineering, University of Pune, India.

Accepted 30 June 2014, Available online 30 July 2014, **Vol.4, No.3 (June 2014)**

Abstract

In Android devices, Memory management has become a major concern because it has significant impact on system performance and battery life. Also it is important to efficiently use and manage the internal and external memory space present inside the mobile operating system. So it is essential to make a facility that helps in reducing memory consumption. The proposed Classic Image Compression Algorithm compress the RGB color image using lossless Image Compression Algorithm with the help of predictive coding based on Color Quantization for Android Mobile Devices. Predictive coding is very effective for lossless image compression. Predictive coding estimates a pixel color value based on the adjacent pixels. To enhance the accuracy of the estimation, we propose a new and simple predictive coding that estimates the pixel color value based on the quantized pixel colors of three neighboring pixels. 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 it will reduce the image size while achieving the best image quality with less data loss. It will display on output screen MSE, PSNR, and compression ratio and compression time of compressed Image.

Keywords: *Color Quantization, Predictive coding, Image Compression, Android devices, Memory Management, Image Representation, Segmentation.*

1. Introduction

Nowadays we are facing the increasing use of images in many parts of our life. 3D vision systems, satellites, cameras, medical equipments etc. All of these equipments use or produce image for different purposes for using these images, for competitive examination we need to upload the compressed image [Fuangfar Pensiri *et el*, 2012]. we have to save or transmit them, 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 several methods for image compression based on the criteria and conditions [Seyed Mehdi Moghadas *et el*, 2011]. Some of these criteria are compression ratio, compression quality, compression time.

Image compression is minimizing the size in bytes of a graphics file without degrading the quality of the image. The reduction in file size allows more images to be stored in a given amount of disk or memory space. It also reduces the time required for images to be sent over the Internet or downloaded from Web pages. This paper is disturbed with lossless compression using the predictive coding for RGB color images. Predictive coding is a compression method used for text and image compression.

It encodes the difference between the current data estimation derived from past data and actual current data to attain more efficient compression [Anupam Mukherjee *et el*, 2012].

To improve the compression efficiency of the predictive coding method, a new estimation method is proposed to give higher estimation accuracy[Y. Sirisathikul *et el*, 2004]. The new method estimates the pixel color value by the average of color values of the neighboring pixels, i.e. the adjacent pixels in the north, northwest, and west directions, with the same quantized color. If all three neighboring pixels have different quantized colors, the estimated value is simply equal to the average of the pixel color values of all three neighboring pixels. It can be shown later that by using the proposed estimation scheme the estimation errors can be reduced significantly leading to an improvement of compression efficiency with MSE, PSNR, Compression Ration and Compression Time. Android supports 3 common image formats PNG, JPEG, GIF. Images are stored in SD card & internal memory. The two most common compressed graphic image formats are the JPEG format and the PNG format in android. In Section 2, we examine literature survey. The proposed is described in Section 3. Section 4 introduces the proposed system architecture. Mathematical Model of the proposed system is presented in Section 5. We conclude the paper in Section 6.

2. Literature Survey

Several studies have been contributed to reduce the memory consumption and to increase the performance of the mobile devices. Image compression is one of the techniques to reduce the memory consumption in android

*Corresponding author: **Rupali Sachin Vairagade**

and helps to increase the performance of the android mobile devices.

In this [Fuangfar Pensiri *et el*, 2012] paper deals with an image compression method using predictive coding which estimates a pixel color value based on the pixel color values of its neighbouring pixels. To enhance the accuracy of the estimation, a new and simple predictive coding that estimates the pixel color value based on the quantized pixel colors of three neighbouring pixels. This prediction scheme can help to minimize the upper bound of the residual errors from the prediction. Experiments were conducted on a set of true color 24-bit images, whose pixel colors were quantized into 2, 4, 8 and 16 colors. The experimental results show that the proposed algorithm outperforms some well known lossless image compression algorithms such as JPEG-LS and PNG by factors of 2-3 in terms of bits per pixel. The results also show that the proposed algorithm gives the best compression rates when colors were quantized into two colors.

In this [Anupam Mukherjee et el, 2012] paper describes a simple and efficient method of color image compression/decompression using the fundamental application of subtraction and formation of a dynamic dictionary after three stages of subtraction. Pixels in different segments generated by a chosen mask window are adjusted such that after three stages of subtraction, either all zero or at most one non-zero element is left. The dictionary stores the coordinates and the value of the non-zero elements. Color transformation that de-correlate color components allow a significant increase in compression ratio achieved by use of standard lossless compression technique applied independently to the each component (RED, GREEN, BLUE). The experimental results show that this algorithm, using the fundamentals of predictive encoding gives a lossless technique of colour image compression. This approach gives better compression ratio than some other lossless techniques and JPEG, JPEG 2000, JPEG LS etc. and the process have 100% PSNR value.

In this [Y. Sirisathikul *et el*, 2004] paper, a simple Color Image Quantization(colormap design) algorithm using squared Euclidean distance of adjacent color points along the highest color variance axis is proposed .The objective of color image quantization is to quantize a true color image, typically $224 \approx 16.8$ million colors for each pixel, into one with fewer colors (256 colors or less). If the image is quantized to 256 colors, the file size will be reduced to one-third of original size. The proposed algorithm is based on a concept that the sum of distances between adjacent colors along the highest color variance axis indicates the dissimilarity of all colors in each color cell. This algorithm is a hierarchically divisive colormap design technique. Colors are sorted along the axis with the highest variance of color distribution.

In this [Marcelo J. Weinberger *et el*, 2000] paper, an image compression method is used which discuss the principles underlying the design of LOCO-I, and its standardization into JPEG-LS. LOCO-I (LOw COmplexity LOssless COmpressionfor Images) is the algorithm at the core of the new ISO/ITU standard for lossless and near-lossless compression of continuous-tone images, JPEG-LS. It is conceived as a low complexity projection of the universal context modelling paradigm, matching its modelling unit to a simple coding unit. By combining simplicity with the compression potential of context models, the algorithm enjoys the best of both worlds. It is based on a simple fixed context model, which approaches the capability of the more complex universal techniques for capturing high-order dependencies. The model is tuned for efficient performance in conjunction with an extended family of Golomb-type codes, which are adaptively chosen, and an embedded alphabet extension for coding of low-entropy image regions. LOCO-I attains compression ratios similar or superior to those obtained with state-of-the-art schemes based on arithmetic coding. Moreover, it is within a few percentage points of the best available compression ratios, at a much lower complexity level.

In this [Hui-yang wang *et el*, 2012] paper, JPEG2000 is transplanted on smart phone based on Android to enhance efficiency of image compression compared with traditional JPEG. JPEG2000, which is wavelet-based, has the features of multi-level image resolution and image quality. JPEG2000 has superior performance and improved the quality of image compression estimated by 30 percent.

In this [Seyed Mehdi Moghadas *et el*, 2011] paper a new quantization method is proposed in this paper which is useful for enhancement of compression quality when each kind of neural network is used to compress the image. By quantizing the image with the proposed method, the numbers of samples which must be reconstructed by neural network is reduced. This causes a remarkable increase in quality of the reconstructed image. For testing the proposed method, auto associative transform coding and by merging it with the proposed quantization method a new compression algorithm is obtained. Results show that the proposed compression algorithm increases the compression quality of the images remarkably. Compression time and complexity in the merged method is also better than JPEG and make it suitable for the systems with low processor and hardware implementation.

In this [Sebastiano Battiato *et el*, 2011] paper Content-aware image resizing are effective algorithms that allow taking into account the visual content of images during the resizing process. The experimental result shows the difference between the iterative approach and non-iterative approach in terms of visual quality and computational time on a mobile device.

In this [Deepalin Kayande *et el*, 2012] paper aims towards giving an approach for better utilization of the internal memory present in Android operating system for mobile phones. For this purpose a dictionary based application called SMSLingo has been developed for SMS texting which requires less memory space and less RAM consumption as compared to default SMS messaging service provided by Android.

3. Proposed Method

1. Color Quantization

Color quantization is a process of dividing a color space of an image into regions. Each region can then be represented by a representative color, normally the centroid of the region. The process can be used to represent a color image using a number of representative colors which take fewer bits to represent. However, this representation scheme introduces image distortion that need to be minimized. The distortion can be measured by the total quantization error which is the sum of squared distances between actual pixel colors and their color representatives. The distance between the color point is measured using Euclidian Distance.

Input: A set I= {(ri,gi,bi) | i=1,2,3…..|I| }

Step 1: It take the color values of each pixel in a region and then the color values of the image are sorted in ascending order based on the color values for each of the color axis i.e. R, G, B color axis of the Region of a image.

Step 2: The highest variance intensity color value is selected to be the principal axis for partitioning the sub region in the current sub regions.

Step 3: Select a Dimension axis with highest variance of color component as a principal axis to become the partitioning points where the principal axis cutting plane pass through.

Step 4: Sort all the colors in ascending order in the region along the principal axis.

Step 5: Consider P_i and P_{i+1} be the to adjacent color points in the sorted list of principal axis,

let, $D_j = d(P_i , P_{i+1})2$ where $d(P_i , P_{i+1})$ is the Euclidean distance between the two adjacent color points.

The distance between a pair of color points can be measured using the Euclidean Distance as follows:

D(Pm,Pn)= $\sqrt{(rm-rn)2+(gm-gn)2+(bm-bn)2}$

Where , the $d(P_m, P_n)$ is the Euclidean distance between the two color points Pm and P_n. r_m,g_m and b_m are the color values of P_m on red, green and blue color axes respectively and r_n,g_n and b_n are the color values of P_n on red, green and blue color axes respectively.

Let dsumi be the sum of squared Euclidean distances of D_1 to D_j as follows:

$$dsum_i = \sum_{j=1}^{i} Dj$$

Step 6: Compute the centroid distance of a region.

$$CentroidDist = \sum_{i=1}^{i} fi \,.\, dsumi / \sum_{i=1}^{n} fi$$

Where, fi is the frequency of ith color and dsumi is the summation of distance between the adjacent colors.

Step 7: Consider sub region I. The current sub region I is partitioned into two smaller sub regions I1 and I2 using the partitioning point division method discussed in step 3 and 4. Each of the three sorted lists, one for each axis is also divided into two lists according to the partitioning point.

Step 8: The process of Partitioning from step 2 to 6 is performed recursively on both I1 and I2 until the number of sub regions equal to the specified value.

2. Color Histogram

A color histogram is a vector where each entry stores the number of pixels of a given color in the image. All images are scaled to contain the same number of pixels before obtaining the histogram, and the colors of the image are mapped into a discrete color space containing n colors.

Typically images are represented in the RGB color space, using a few of the most significant bits per color channel to discrete the space. Color histograms are widely used for content-based image retrieval because they are trivial to compute, and despite their simplicity, exhibit attractive properties. Since color histograms do not relate spatial information with the pixels of a given color, they are largely invariant to the rotation and translation of objects in the image. In proposed work it will consider the number of pixels of a Red, Green, Blue color in the images. All the count of same color it will store in a vector.

3. Predictive Coding

Let, the output from the color quantization it will give a centroid of the entire region. The predicted color value of each pixel, one at a time, starting from the leftmost column to the rightmost column and from the top row to the bottom row, based on the quantized colors of their adjacent pixels.

Input_Predicate = {Image centroid, Ht_image , Wt_image}

Let, the predicate color method of the proposed algorithm gives the value of predicate color depends upon the location of the current pixel in the current region e. g. P(i, j). Consider the three adjacent pixel of the current pixels N, NW and W, of the current pixel P. let Pc, Nc, NWc and Wc represent the pixel color values of P, N, NW and W respectively. By identifying the color value, residual error can be calculated and the output of the Predictive coding is:

Output = { Predicate_color, Residual error}

4. Encoding Process

The Proposed encoding process of image compression consists of three main tasks:

1. Color Quantization
2. Predictive Coding
3. Encoding process using Huffman Coding

In color quantization the number of regions on given image discussed above and the output of color quantization task returns the centroid of each region. In Predictive coding it will calculate the predicate color value of each pixel based on the quantized color of adjacent pixel. In the last task it encodes the residual error with other parameter as number of centroid and the value of centroid using Huffman coding.

Consider, a set of centroid C= { C_1, C_2, C_3,……..C_n} with the intensity value of a centroid f(a_i). Calculate the Prefix value for centroid of a region to be encoded using Huffman Coding.The value of intensity A for C that minimizes number of bits:

$$\mathbf{B\ (I)} = \sum_{c=1}^{n} f(Ci)\ \ L(A(Ci))$$

Where, It is used to encode a intensity value and A(C_i) is the codeword for encoding C_i , L(A(C_i) is the length of the codeword A (C_i)

4. Proposed System Architecture

In proposed System architecture for Image Compression Algorithm using predictive coding based on Color Quantization for Android Mobile Devices, it will run android application on any mobile device where the android operating system is present. Its takes a Input as Images for application it will take Images from Gallery of Mobile Phones, Image captured from Camera and Images downloaded from WAP or Internet. Either select single image or multiple Images for Image compression, for Single Image it will show the file size of the Image i.e original image stored in mobile device, and Extension or type of the image for ex. .jpeg or .png. Our objective is to enhance the reduction of memory consumption in android mobile devices with Image Compression Algorithm using predictive coding based on Color Quantization technique. It will take original image and it stores all the RGB values of images into a Matrix then Color quantization is performed 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 pixels value.

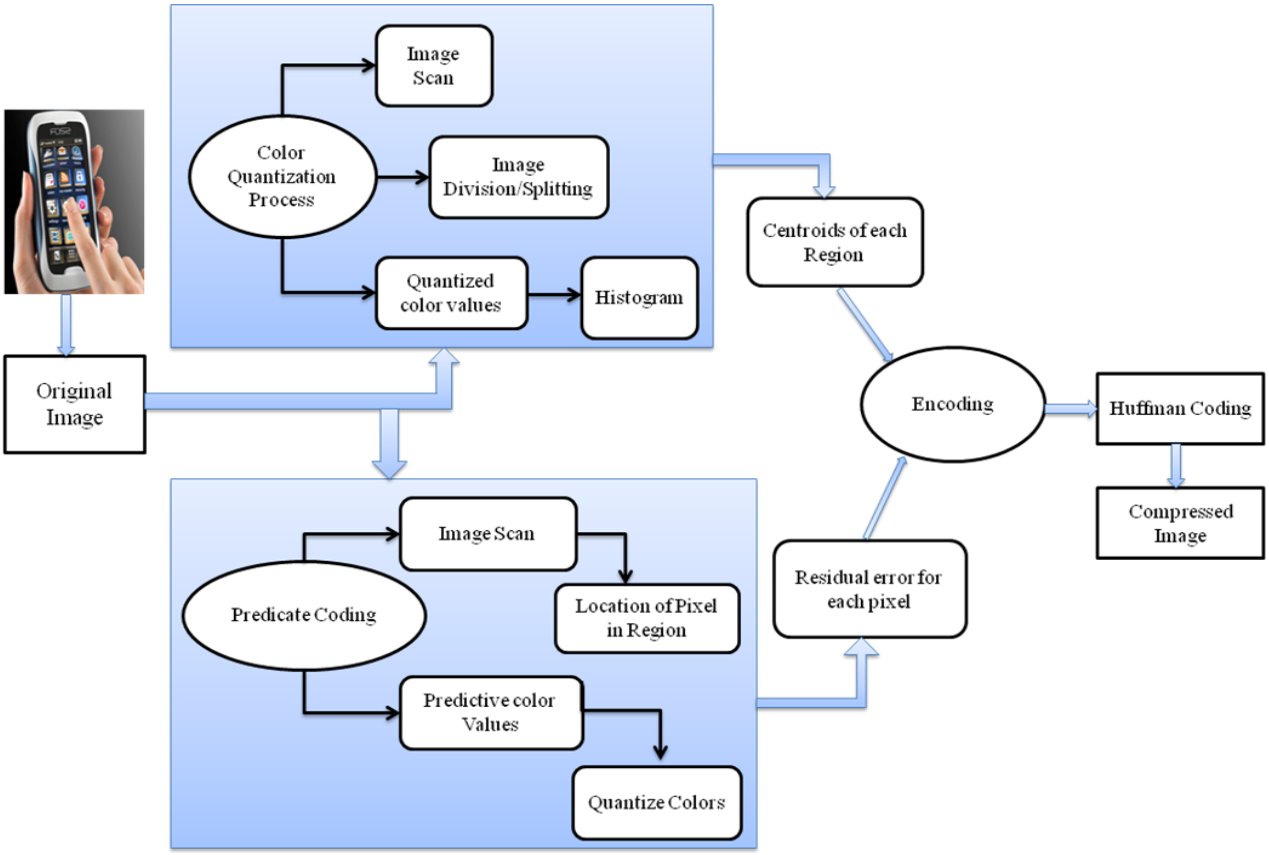

Figure 1.Proposed System Architecture

The predicted color value of each pixel, one at a time, starting from the leftmost column to the rightmost column and from the top row to the bottom row, based on the quantized colors of their adjacent pixels. The residual error of each pixel is computed which is the the longest distance between any pair of color points of the given image without color quantization. The last task encodes the residual errors as well as some parameters, such as the number of centroids and the values of the centroids, using the Huffman Coding method. After encoding we get compressed image as an output. The results shown by this application is in the form of MSE, PNSR Ration, Compression ratio and Compression time.

5. Mathematical Model

Let S be the System such that,

S = {Input, DD, NDD, SI, RI, Image Compression, Sc,

CR, Mt, Mavi, Macc, Succ, Failure, Output}
Input = {P_1, P_2, P_3} = {Number of Images from Gallery, Camera & WAP}
P_1= {Image Capture from Camera type is .jpeg or .png}
P_2= {Image is Downloaded from web in Mobile Devices type is .jpeg or .png}
P_3= {Image is already present in a Gallery in jpeg or png}
DD = Deterministic Data
= CI ϵ AI
Where, CI = Required Images requested by client to compress = {P_1, P_2, P_3}
AI = Available Images in Mobile Device and image is divided into a region.
NDD = Non-Deterministic Data= {P1= ϕ, P2= ϕ, P3= ϕ}
= CI ϵ AI
= Required Images resources are not available.
SI = {Size, Resolution, DPI, Image Type}
= {Single Image Properties from P1, P2, P3,…….}
RI = {SI is divided into number of regions }
= {R_1, R_2, R_3, ……..R_n}
For each region of R_1, Let Pixel are x_1, x_2, x_3, x_4 and corresponding pixel intensities { $f_R(x_1)$, $f_G(x_1)$, $f_B(x_1)$ }, { $f_R(x_2)$, $f_G(x_2)$, $f_B(x_2)$ }, { $f_R(x_3)$, $f_G(x_3)$, $f_B(x_3)$ }, { $f_R(x_4)$, $f_G(x_4)$, $f_B(x_4)$ }, From that select the highest intensity value of region R_1 for each component RGB say { $f_R(x_3)$, $f_G(x_3)$, $f_B(x_3)$ }

Image_Compression={Color Quantization, Predicate_color, Encoding }

Color_Quantization

A color quantization could work as the first step for image compression. A true color image can be defined as follows: $F: M \times N \rightarrow C \subseteq \Psi$,
Where, Ψ = { (ri,gi,bi) | $0 \leq$ r, g, b ≤ 255 } is the RGB color space, (x, y) $\in$ M $\times$ N are the co-ordinate of a pixel, M and N is the integer set of colors used in the image is C = { C_1,C_2, ----C_n }. There are $256 \times 256 \times 256$ possible combination of red green and blue components. A quantized image may be regarded as mapping defined by:

$Q: M \times N \rightarrow R \subseteq \Psi$

Where, R= {r_1, r_2, …….r_k} is a set of representative color used in the quantized image. It will consider the RGB color image is divided into the number of regions and each region is represented by representative color:
Input: A set I= {(ri,gi,bi) | i=1,2,3…..|I| } ={Input original Image}

Output = {Centroid of the region}

Predicate color

Input_predicate = {Image centroid, Ht_image, Wt_image}

The predicate color method uses the following conditions to determine the color value of P and P':

1. If the current pixel is at leftmost top position then there is no prediction of color will occurs. In that case there is no adjacent pixel of current pixel so the value of predicate color P' = 0
2. If the current pixel is at first row not in the top leftmost corner then in this case value of predicate color P' = Wc. Wc is the intensity value of the W pixel.
3. If the current pixel is at first column not in the top leftmost corner then in this case value of predicate color P' = Nc. Nc is the intensity value of the N pixel.
4. If the current pixel P has all the three adjacent pixel and at least two of the three adjacent pixels have their color located in same quantized color regions then P'= the average value of color value of these pixel. Otherwise, the three adjacent pixels located in different color regions then P'= the average value of color value of three pixels. To search which quantized color region a pixel color belongs to it has nearest pixel from the centroid.

Output_Predicate = {Predicate_color, Residual error}

Encoding

Input = {Centroid, Residual_error}

Consider, a set of centroid C= { C_1, C_2, C_3,……..C_n} with the intensity value of a centroid f(ai). Calculate the Prefix value for centroid of a region to be encoded using Huffman Coding.

Output = {It will provide a Compressed Image}

CR = Compression Ratio = Original Image / Compressed Image

Mt = {1……n} = Total Memory

Mavi = {1…….n} = Available Memory

Mocc = {1……m} = Occupied Memory

$$Mavi = \sum_{i=1}^{n} M - \sum_{i=1}^{m} M_{occ}$$

Output :{ Image is compressed and Save the more Memory of Android Devices.}

Success: {Reduced the file size of Image & reduced consumption of memory.}

Failure: {The File size of the image is not reduced & more consumption of memory.}

5. Experimental Results

The Classic Image Compression Application for android Mobile devices is tested on 200 images of type .jpeg and .png using predictive coding based on color quantization. Application is tested on different images varying the

image size from 50kb to 5Mb. During Testing, image type is properly checked to avoid false results and failure of the system. Image must be in .jpeg or .png format only. For detailed analysis and to check the performance of the image, various performance parameters have been computed. The performance parameters include Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR), Compression Ratio and Compression Time. These parameters are computed by Classic Image compression Application on Android Mobile Devices.

Test Images for Classic Image Compression Application

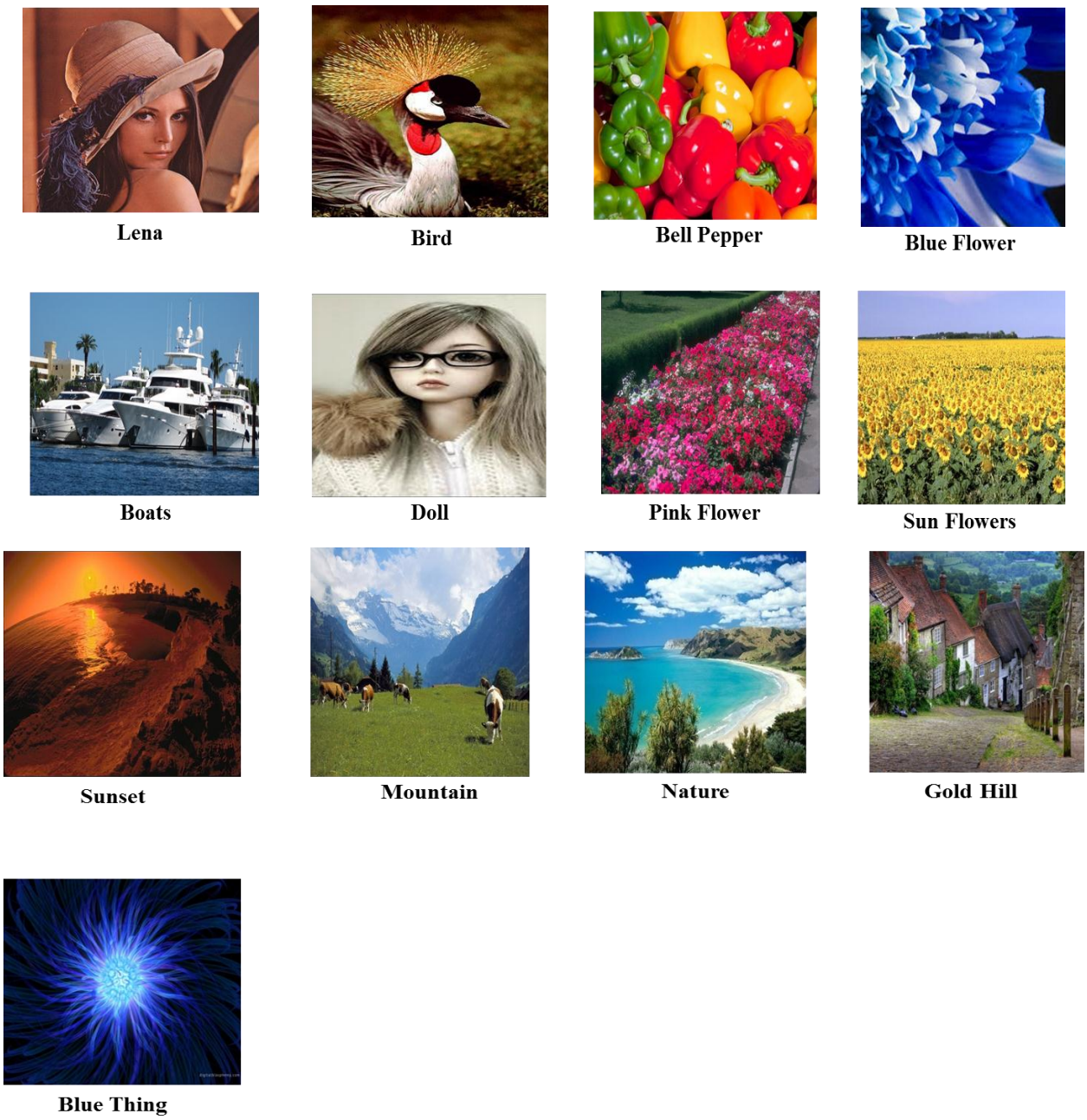

Figure 2 Difference original Image and Compressed Image

By observing the results and analyzing it, we can determine how efficient an application is working and its performance with respect to other existing image compression tool.

The below table shows the results in terms of Compressed Size, MSE, PSNR and No. of Pixels of the compressed image.

Important parameter in Image compression is Compression Ratio and quality of Image. We have tested using android mobile devices the result of image compression using two parameter compression ratio and compression time.

Computation of Compression Ratio is:

Compression Ratio = Original Image / Compressed Image

Table 1 Compression Ratio and Compression time of test Images

Sr. No	Image Name	Image Type	Original size (kb)	Compressed size (kb)	MSE	PSNR	No. of pixels
1	Lena	Png	606	462	79.245	29.141	262144
2	Bird	jpeg	340	21.5	175.176	25.69	485006
3	Bell peppers	Jpeg	59	10.1	36.923	32.457	307200
4	Boat	Jpeg	75	9.6	227.068	24.569	209920
5	Doll	Png	110	80	205.146	25.01	50440
6	Pink flower	png	931	800	333.215	22.903	370757
7	Blue flower	Png	138	97	173.817	25.729	50451
8	Sun flowers	Jpeg	317	24	301.754	23.334	402432
9	Sunset	Jpeg	159	10.5	60.738	30.296	369768
10	Mountain	Jpeg	229	12.6	216.21	24.782	392448
11	Nature	Png	883	746.2	311.211	23.2	407835
12	Gold hill	Jpeg	157	21	313.963	23.162	426400
13	Blue thing	Jpeg	138	30.1	108.671	27.769	995328

The below table shows the results in terms of Compression Ratio and Compression Time using the test images

Sr. No	Image Name	Image Resolution	Compression Ratio	Compression Time(Sec)
1	Lena	512 x 512	1.217	10
2	Bird	562 x 863	1.083	18
3	Bell peppers	640 x 480	0.765	11
4	Boat	640 x 328	0.9	7
5	Doll	194 x 260	0.955	1
6	Pink flower	743 x 499	0.651	15
7	Blue flower	251 x 201	0.6	1
8	Sun flowers	768 x 512	0.948	16
9	Sunset	744 x 497	0.825	13
10	Mountain	768 x 511	0.851	15
11	Nature	795 x 513	0.958	16
12	Gold hill	800 x 533	1.028	16
13	Blue thing	1152 x 864	1.198	43

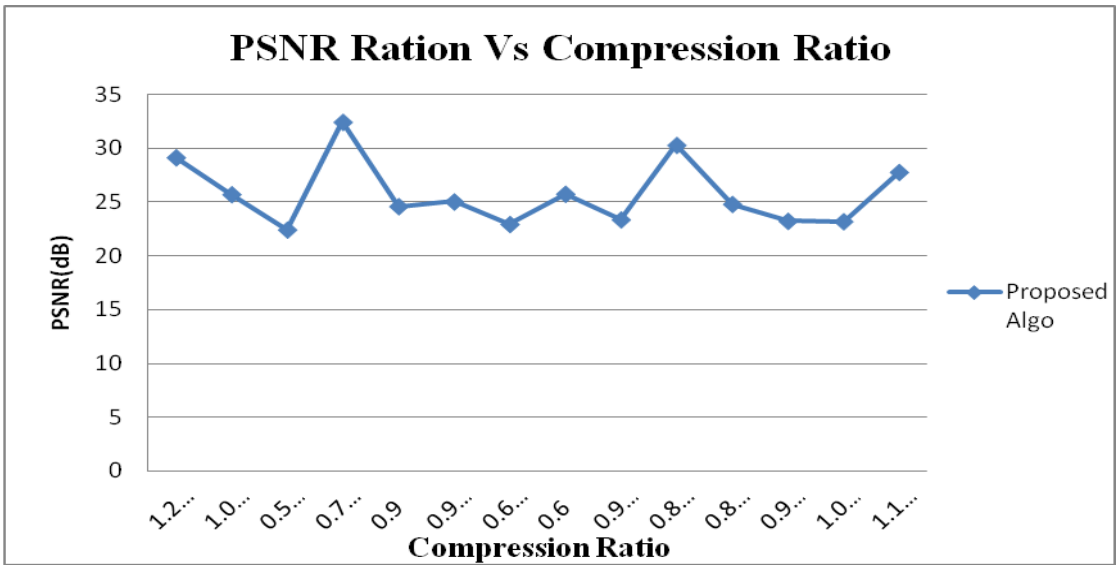

Figure 3. Graphical Representation of Compression Ratio vs. PSNR

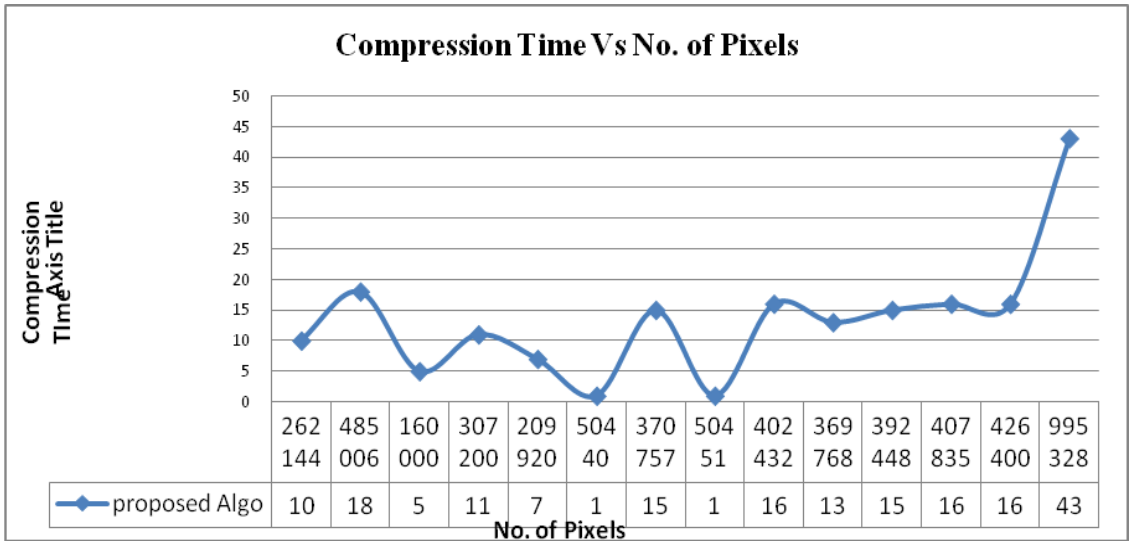

Figure 4. Graphical Representation of Compression Time vs. No. of Pixels

Conclusion

We proposed Android classic Image Compression Application. In proposed system first scan the RGB color image to get all the color values of Image and it get stored in color matrix then compute centroid of the region using color quantization process. We then compute residual error of the neighboring pixels values using predictive coding method. We used the centroid of the each regions and residual error values for encoding, Encoding process is done using Huffman coding to get bits of stream for image pixels values as compressed image. We tested an application on different set of images.

The application is also efficient since it works to reduce the memory consumption for android mobile devices. By using this Classic Image Compression Application, a compressed image is displayed on mobile screen, obtained a good quality image with minimum data loss. It takes minimum bandwidth to transfer image on the network. We conclude that, this application gives good results with high compression ratio, less compression time and maintain good quality of image than other existing image compression Application.

References

Fuangfar Pensiri, Surapong Auwatanamongkol(2012), A lossless image compression algorithm using predictive coding based on quantized colors, *Department of Computer Science, School of Applied Statistics,National Institute of Development Administration118, Seri Thai Road, Bangkok 10240 Thailand*, Issue 2, Volume 8.

Anupam Mukherjee, Mitankar Das Sarkar, Amiya Halder(2012), Predictive Lossless Color Image Compression using Arithmetic Operation, International *Journal of Computer Applications (0975 – 8887)*,Volume 43,Pagr No.5.

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.

Marcelo J. Weinberger, Gadiel Seroussi, and Guillermo Sapiro(2000),The LOCO-I Lossless Image Compression Algorithm: Principles and Standardization into JPEG-LS, *IEEE Transactions On Image Processing*, Vol. 9, No. 8.

Hui-yang wang, xiang-dong(2012),Study of jpeg2000 on Android, *School of Information and Communication Engineering, Beijing University of Posts and Telecommunications*, proceedings of IEEE

Seyed Mehdi Moghadas, Hossein Pourghasem(2011), New Image Compression Algorithm using Proposed Quantization Approach, *2011 International Conference on Pattern Analysis and Intelligent Robotics 28-29, Putrajaya, Malaysia.*

Sebastiano Battiato, Giovanni Maria Farinella, Nicol`o Grippaldi, Giovanni Puglisi, Content-based image resizing on mobile devices, Image Processing Laboratory,Department of Mathematics and Computer Science,University of Catania, Italy.

DeepalinKayande and urmila shrawankar(2012), Performance Analysis for Improved RAM utilization for Android Application, *Software Engineering (CONSEG), 2012 CSI Sixth International Conference*.

M. J. Weinberger, G. Seroussi, and G. Sapiro(1996), LOCO-I: A low complexity, context-based, lossless image compression algorithm, *in Proc. 1996 Data Compression Conference, Snowbird, UT*, pp. 140–149.

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 .

A. J. R. Neves and A. J. Pinho (2006), A Bit-Plane Approach for Lossless Compression of Color-Quantized Images , *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 429-432.

C. Chang, P. Xu, R. Xiao, and T. Srikanthan(2005), New adaptive color quantization method based on self-organizing maps, *IEEE Transactions on Neural Networks*, Vol. 16. No. 1, pp. 237-249.

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

http://en.wikipedia.org/wiki/Image_file_formats

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

http://developer.android.com/guide/practices/screens_support.html

http://www.serif.com/appresources/HPX4/Tutorials/engb/photoplus_tutorials/photo_image_size.htm

http://www.ccbirding.com/dcsig/howto/dpi/dpi.pdf

http://www.webdesignerdepot.com/2010/02/the-myth-of-dpi/

http://argillander.wordpress.com/2011/11/24/scale-image-into-imageview-then-resize-imageview-to-match-the-image/#comment-52

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

http://searchciomidmarket.techtarget.com/definition/image-compression

http://en.wikipedia.org/wiki/Image_scaling