ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 12, December 2015

Predictive Coding: A Reducing Memory Consumption with a Lossless Image Compression Algorithm

Rupali Sachin Vairagade, Yogita S. Hande, Sandeep M. Chitalkar, Santosh S. Darade

Assistant Professor, Dept. of Computer Engineering, SITS, Savitribai Phule Pune University, Pune, India

Assistant Professor, Dept. of Computer Engineering, SITS, Savitribai Phule Pune University, Pune, India

Assistant Professor, Dept. of Computer Engineering, SITS, Savitribai Phule Pune University, Pune, India

Assistant Professor, Dept. of Computer Engineering, SITS, Savitribai Phule Pune University, Pune, India

ABSTRACT: This paper describes a simple but effective predictive coding technique for color image compression using color quantization. In Android mobile devices, Memory management has become a major apprehension because it has significant impact on performance of system and battery life. Also it is important to efficiently use and manage the internal and external memory space available inside the mobile operating system. Efficient memory consumption plays an important role in better performance of the device. So it is essential to make a facility that helps in reducing memory consumption. The proposed Android Image Compression Tool compress the color image using lossless Image Compression Algorithm using predictive coding built on Color Quantization for Android Mobile Devices. The objective of image compression is to shrink the redundancy of the image and to store or transmit data in an efficient form. The experimental results reveal that the algorithm is effective and yields a better performance time when compared with other existing mobile applications. The key goal of such system is to decrease the storage quantity as much as possible and encoded image displayed in the output screen that will be similar to the original image. This proposed system will reduce the image size while achieving the best image quality with lesser data loss.

KEYWORDS: Color Quantization, Predictive coding, Android OS, Memory management, Image Compression.

I. Introduction

Due to high involvement in social networking applications, there is high consumption of memory to run these applications and very limited amount of memory remain available for android devices. Due to this reason there is necessary to work on the memory consumption of mobile devices. The image compression on android mobile devices is new and emerging field with many challenges due to limited hardware and connectivity. So image compression tool helps to reduce the consumption of expensive resources such as memory storage (External and Internal) and transmission cost due to low bandwidth requirements. The efficient memory management plays an important role for the better performance of the mobile devices. Image compression is process of data compression that encodes the original image with limited bits. The objective of image compression is reduction in file size allows more images to be kept 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.

Nowadays we are facing the increasing use of images in many parts of our life e. g. 3D visualization systems, satellites, cameras, medical equipments etc. All of these equipments use or produce image for different purposes for using these images, for competitive investigation we need to upload the compressed image [1]. we have to save or communicate them, then because of the restriction in disk space and channel bandwidth We almost always need image compression for decreasing the size of data which must be save or transmit[1]. There are several methods for image compression based on the standards and conditions. Some of these criteria are compression ratio, compression quality, compression time. Image compression is decreasing the size in bytes of a graphics file without degrading the quality of the image. The compression in file size allows more images to be stored in a given amount of disk or memory space. It also decreases the time required for images to be sent over the Internet or downloaded from Web pages [2].

DOI: 10.15680/IJIRCCE.2015. 0312053

In today's world, mobile phones provide a wide variety of functionalities that are not always related to calling or sending messages. Those functionalities include web browsing, playing games or music, banking, taking photos and so much more. Classic Image compression is an android mobile application that aims to allow the user to compress the images selected from gallery, camera and internet. It is an application that compress image with good compression ratio and good quality of image. Also it will also calculate the compression time. The major issue of android mobile devices was low memory space, by using classic image compression application it will help to reduce the memory consumption of android mobile devices.

The predictive coding has proven to be effective for image compression. Predictive coding estimate a pixel color values of its neighbouring pixel. To enhance the accuracy of the estimation, we propose a predictive coding that estimates the pixel color value based on the quantized pixel colors of the three neighbouring pixels. Android classic image compression application also help to reduce the cost incurred in transferring the image on the web and social media. Another motivational driver is the less availability of image compression application that will help to compress the image with less data loss and best retention of image quality.

II. Related work

Sr. no	Author	Techniques	Pros	Cons
1	Fuangfar Pensiri, Surapong Auwatanamongkol	Image Compression technique using Predictive Coding	Predictive coding that estimates the pixel color value based on the quantized pixel colors of three neighbouring pixels	Experiments were conducted on a set of true color 24-bit images, whose pixel colors were quantized into 2, 4, 8 and 16 colors.
2	Y. Sirisathitkul, S. Auwatanamongkol , B. Uyyanonvara	colormap design technique for color image quantization.	As a result, less execution time while produces relatively low quantization errors.	Division of Image will loss the quality of image
3	Anupam Mukherjee, Mitankar Das Sarkar, Amiya Halder	Predictive Lossless Color Image Compression using Arithmetic Operation	This approach gives better compression ratio and the process has 100% PSNR value.	if four pixel are of different intensity value, then after consecutive maximum three subtraction the block will be one non-zero value or all are zero.

III. Proposed algorithm

To enhance the reduction of memory consumption in android mobile devices using Image Compression Algorithm with use of predictive coding based on Color Quantization Method.

A. **Algorithm 1 (Color Quantization):**

- Step 1: Consider the original Image and scan the image to identify how many pixels are present in the image.
- Step 2: Convert original image into 24-bit color Bitmap Image.
- Step 3: Store all color intensity values of image into arraylist.
- Step 4: Consider RGB color values of each pixel and store all RGB color pixels values into a arraylist.
- Step 5: Compute the Histogram Matrix.
- Step 6: Sort all the RGB pixels in the ascending order.
- Step 7: Select the highest variance of color Component to become a partitioning point.

DOI: 10.15680/IJIRCCE.2015. 0312053

- Step 8: Calculate a Distance between a pair of color Points can be measured using a Euclidian distance. Where, the d(Pm, Pn) is the Euclidean distance between the two color points Pm and Pn. rm,gm and bm are the color values of Pm and rn, gn and bn are the color values of Pn on red, green and blue color axes.

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

- Step 9: Set the value of k for color quantization. The clustering-based algorithms perform clustering the color space into K-desired clusters. So image is divided into number of partitions.
- Step 10: Compute the sum of squared of Euclidian distance of colors from D1 to Di.

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

- Step 11: Compute the Centroid Distance of Each 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 12: The partitioning process from step 4 to 10 is performed until the number of sub regions equals to a specified value of k.
- Step 13: End

B. Algorithm 2 (Color Quantization):

- Step 1: Consider the original Image and scan the image to identify how many pixels are present in the image.
- Step 2: Convert original image into 24-bit color Bitmap Image.
- Step 3: Store all color intensity values of image into arraylist.
- Step 4: Consider RGB color values of each pixel and store all RGB pixels color values into a Arraylist.
- Step 5: A color histogram is a vector where each entry stores the number of pixels of a given color in the image.
- Step 6: All the RGB count of repeated colors Stored into Arraylist.
- Step 7: Store the value of Histogram Matrix.

C. Algorithm 3 (Predictive Coding)

- Step 1: Consider Original Image
- Step 2: Consider the Current pixel X from the image with their corresponding RGB values.
- Step 3: Consider three adjacent pixels N, NW, W of the current Pixel X, with the RGB color intensity values of all adjacent pixels.

N	*NW*
W	*X*

- Step 4: Compute the predicate value of pixel X, So that consider the conditions for current pixel X.
- Step 5: if the Pixel X at the leftmost top position then
  X’ = 0 Value of predicate color
- Step 6: else if the pixel X at first row not in the top leftmost corner then
  X’ = Wc Value of predicate color
- Step 7: else if the pixel X at first column not in the top leftmost corner then
  X’ = Nc Value of predicate color
- Step 8: else if the current pixel X has all three adjacent pixel from that atleast two of the three adjacent pixels have their RGB colors located in the same region then
  X’ = the average value of these RGB pixels
- Step 9: else if the current pixel X has all three adjacent pixel have their RGB colors located in the different region then
  X’ = the Average value of three RGB pixels

- Step 10: Optimal number of colors give the best compression rate for image.
- Step 11: Stop

D. Algorithm 4 (Encoding Process)

- Step 1: Take the input from Predictive coding.
- Step2: Discard all the duplicate pixels present in RGB color Values depending upon the values present in either same or different region.
- Step 3: Encode the intensity value of Image to give binary code of the image.
- Step 4: Encode the Data Stream of Original Image.
- Step 5: Store and Display the Compressed Image on device.
- Step 6: Compute Compression Ratio, Mean Square Error and Peak Signal to Noise Ratio.

E. Calculation of MSE, PSNR and CR

- CR = Compression Ratio = Original Image / Compressed Image
- MSE:

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} d(c[i,j], q(c[i,j]))2$$

Where c[i, j] and q[i, j] are original pixel and quantized pixel respectively, MN total number of Pixel in the image (M columns and N rows) and d(x, y) is Euclidian distance between the color x and color y.

- PSNR:

$$\mathrm{PSNR} = 10\,.\log_{10}\left(\frac{\mathrm{MAX} * \mathrm{MAX}}{\mathrm{MSE}}\right)$$

Where, MAX is the maximum possible pixel value of the image and MSE is mean Square Error.

IV. 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 algorithm.

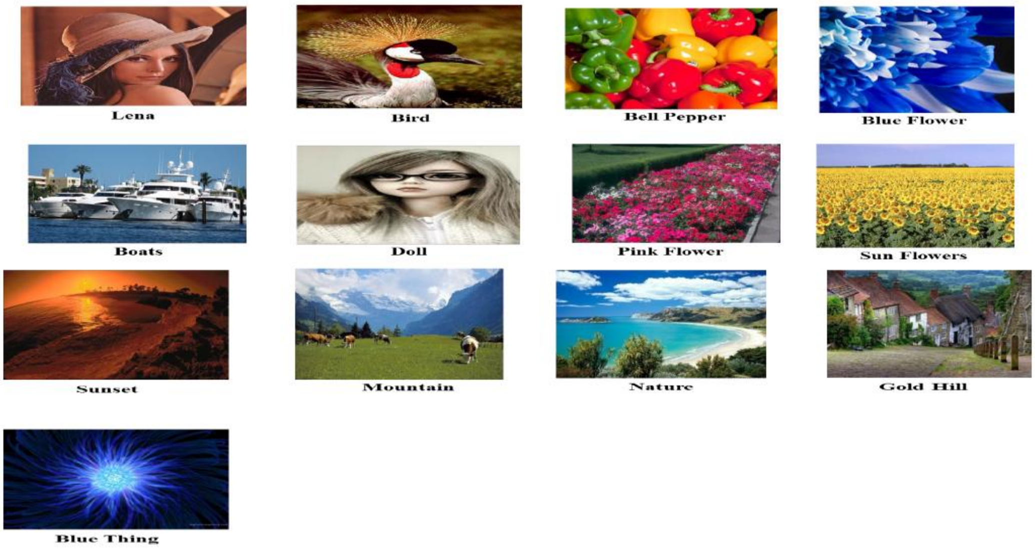


Fig 1. Test images for image compression

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.

Table 1: Below table shows differenceof Original Image and Compressed Image

Sr. no	Image Name	Image type	Original Size (KB)	Compressed size (KB)	MSE	PSNR	No. of Pixel
1	Lena	Png	606	462	79.2	29.141	262144
2	Bird	jpeg	340	21.5	175	25.69	485006
3	Bell peppers	Jpeg	59	10.1	36.9	32.457	307200
4	Boat	Jpeg	75	9.6	227	24.569	209920
5	Doll	Png	110	80	205.14	25.010	50440
6	Pink flower	png	931	800	333.21	22.903	370757
8	Blue flower	Png	138	97	173.81	25.729	50451
9	Sun flowers	Jpeg	317	24	301.75	23.334	402432
10	Sunset	Jpeg	159	10.5	60.738	30.296	369768
11	Mountain	Jpeg	229	12.6	216.21	24.782	392448
12	Nature	Png	883	746.2	311.21	23.20	407835
13	Gold hill	Jpeg	157	21	313.96	23.162	426400
14	Blue thing	Jpeg	418	30.1	108.67	27.769	995328

This table shows the results of multiple images using the image resolution indicating size of images. It will show the comparison of compression ratio and compression time. For 1 Mb size of image it will take only 10 sec time for compression.

Table 2: The below table shows the results in terms of Compression Ratio and Compression Time of test images

Sr. No.	Image Name	Image Resolution	Compression Ratio	Compression Time
1	Lena	512 x 512	0.948	10
2	Bird	562 x 863	0.595	19
3	Bell peppers	640 x 480	0.765	11
4	Boat	640 x 328	0.90	7
5	Doll	194 x 260	0.955	1
6	Pink flower	743 x 499	0.651	15
7	Blue flower	251 x 201	0.60	1
8	Sun flowers	768 x 512	0.948	16

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 12, December 2015

10	Sunset	744 x 497	0.825	13
11	Mountain	768 x 511	0.851	15
12	Nature	795 x 513	0.958	16
13	Gold hill	800 x 533	0.567	17
14	Blue thing	1152 x 864	0.663	43

This below table shows the results of different image compression tool existing at play store. It will show the comparison of different parameter (Image Name, original image, compressed size and compression ratio) for existing android application and proposed classic image compression algorithm.

Table 3: Shows Compression of Existing Android Application and Proposed Application

Sr. No.	App Name	Image name	Original Image (kb)	Compressed Size (kb)	Compression Ratio
1	Img Compress Free	Lena	606	25	75.76
		Bird	340	61.8	94.50
		Bell pepper	59	24.7	97.61
		Doll	110	7.6	85.53
2	Rabido	Lena	606	108	94.39
		Bird	340	11	69.09
		Bell pepper	59	6	90.17
		Doll	110	25	95.60
3	Hipix Plus	Lena	606	76	92.03
		Bird	340	55	93.82
		Bell pepper	59	21	97.19
		Doll	110	36	96.94
4	Reduce Photo Size	Lena	606	43	85.91
		Bird	340	92	96.30
		Bell pepper	59	35	98.31
		Doll	110	5	78.00
5	Image Optimizer	Lena	606	443	98.63

DOI: 10.15680/IJIRCCE.2015. 0312053

		Bird	340	67.5	94.96
		Bell pepper	59	26	97.73
		Doll	110	73.49	98.50
6	Classic Image Compression (Proposed)	Lena	606	462	98.69
		Bird	340	21.5	84.19
		Bell pepper	59	10.1	94.16
		Doll	110	80	98.63

The below table show the results of different android image compression application. It will show the comparison of different parameter (Android application size, Compression ratio, Display compression time, and image quality and memory optimization) for existing android application and proposed classic image compression algorithm. The size of proposed classic image compression compare to other application is very less, it will not reduce the quality of image.

Table 4: The table shows the Android Application comparison on the basis of performance parameter

Parameters **Application Name**	**Application Size**	**Compression Ratio %(Approx.)**	**Display Compression Time**	**Image Quality**	**Memory Optimization**
Img Compress Free	866 Kb	50	No	Yes	Yes
Rabido	830Kb	40	No	Yes	Yes
Hipix Plus	1.4Mb	50	No	Yes	Yes
Reduce Photo Size	Varies with device	50	No	Yes	Yes
Image optimizer	1.6 Mb	60	No	Yes	No
Proposed Classic Image Compression	710 Kb	70	Yes	Yes	Yes

The below figure show the graphical representation of compression ratio of images and PSNR Ratio, Second graph will show the comparison of compression time and no. of pixel. It will take less time for high resolution images. Third graph shows graphical representation of existing android application and compression ratio.

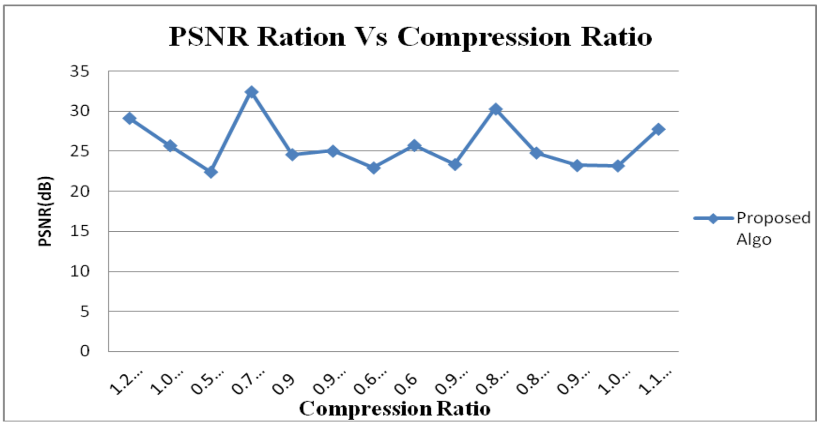
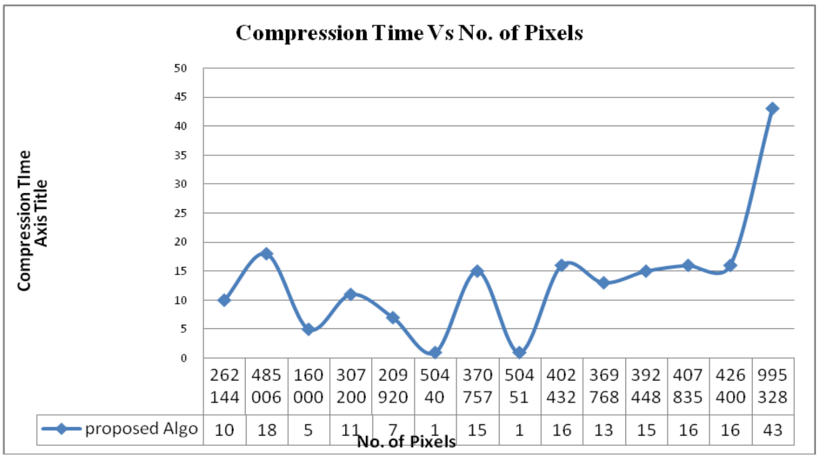


Fig. 2: Graphical Representation of Compression Ratio vs. PSNR



	262 144	485 006	160 000	307 200	209 920	504 40	370 757	504 51	402 432	369 768	392 448	407 835	426 400	995 328
proposed Algo	10	18	5	11	7	1	15	1	16	13	15	16	16	43

Fig. 3: Graphical Representation of Compression Time vs. No. of Pixels.

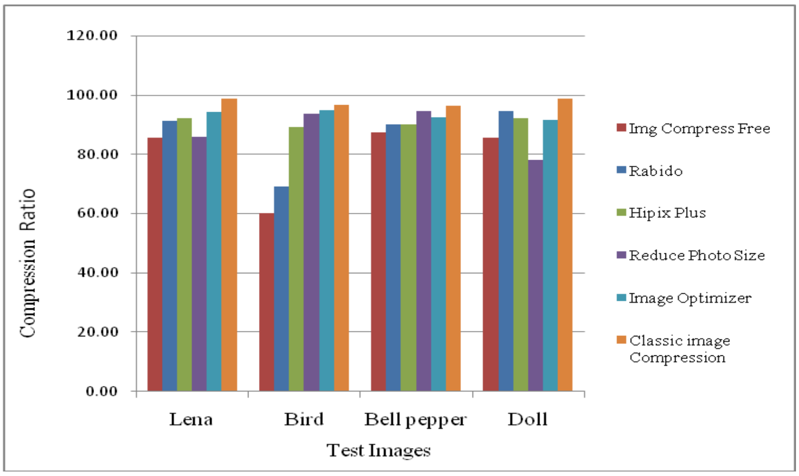


Fig. 4: Graphical Representation of Existing Android Application vs. Compression Ratio

V. CONCLUSION AND FUTURE WORK

Android operating system has the major issue of memory management, classic Image Compression Tool will help you to overcome the issues of memory management. We 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 neighbouring 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 of android mobile devices. By using this Classic image compression application, a compressed image is displayed. obtained a good quality image with minimum data loss. It takes minimum bandwidth to transfer image on the network.

REFERENCES

1. Fuangfar Pensiri, Surapong Auwatanamongkol, “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.
2. Anupam Mukherjee, Mitankar Das Sarkar, Amiya Halder, “Predictive Lossless Color Image Compression using Arithmetic Operation”, *International Journal of Computer Applications* (0975 – 8887),Volume 43– No.5, April 2012.
3. Y. Sirisathikul, S. Auwatanamongkol, and B. Uyyanonvara, “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, 2004, pp. 1025-1043.
4. Marcelo J. Weinberger, Gadiel Seroussi, and Guillermo Sapiro, " The LOCO-I Lossless Image Compression Algorithm:Principles and Standardization into JPEG-LS, IEEE Transactions On Image Processing, VOL. 9, NO. 8, August 2000.
5. Hui-yang wang, xiang-dong, ”Study of jpeg2000 on Android” , School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, proceedings of IEEE, 2012..
6. Seyed Mehdi Moghadas, Hossein Pourghasem , “New Image Compression Algorithm using Proposed Quantization Approach”, 2011 International Conference on Pattern Analysis and Intelligent Robotics 28-29 June 2011, Putrajaya, Malaysia.
7. 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.
8. DeepalinKayande and urmila shrawankar , “Performance Analysis for Improved RAM utilization for Android Application”, Software Engineering (CONSEG), 2012 CSI Sixth International Conference.
9. M. J. Weinberger, G. Seroussi, and G. Sapiro, “LOCO-I: A low complexity, context-based, lossless image compression algorithm,” in Proc. 1996 Data Compression Conference, Snowbird, UT, Mar. 1996, pp. 140–149.
10. R. N. Neelamani, R. Queiroz, Z. Fan, and R. Baraniuk, JPEG Compression History Estimation for Color Images, IEEE Transactions on image processing, Vol. 15 No. 6, 2006, pp. 1365-1378 .
11. A. J. R. Neves and A. J. Pinho, A Bit-Plane Approach for Lossless Compression of Color-Quantized Images , IEEE International Conference on Acoustics, Speech and Signal Processing, 2006, pp. 429-432.
12. C. Chang, P. Xu, R. Xiao, and T. Srikanthan, New adaptive color quantization method based on self-organizing maps, IEEE Transactions on Neural Networks, Vol. 16. No. 1, 2005, pp. 237-249.