A Pixel Count Approach for Lossy Image Compression

Tanuja R. Patil and Vishwanath P. Baligar

Abstract Digital image storage and transmission plays a very important role in today's modern world, as most of the data transfer involves images. Hence, digital image compression is of great importance. The compression leads to either lossy or lossless type of images. Here, we discuss about a unique approach for lossy image compression, which involves a threshold. The number of pixels whose sum is lesser than a threshold is counted, and this count is saved in a file instead of actual pixel intensity values. A difference between the threshold and the computed sum is also stored. Later, reconstruction is done by reading the count and difference values. Average is calculated, and the count number of pixels is replaced with this value. We find that we can achieve better quality at lower PSNR values with this approach as compared to JPEG algorithm.

Keywords Low PSNR · Lossy image compression · Threshold · Comparison with JPEG

1 Introduction

Digital images are inevitable in the transfer of information nowadays. They require large amount of memory for storage as well as for transmission, and also, time consumed to transmit is very high. Hence, it is very much essential that images are to be compressed. Some applications require lossless compression such as medical imaging, satellite imagery, but some applications like multimedia and GIS prefer highly compressed data rather than high-quality images. Based on these needs, we have lossless and lossy type of image compression techniques.

T. R. Patil (✉) · V. P. Baligar
K.L.E. Technological University, Hubballi, India
e-mail: tanuja_p@kletech.ac.in

V. P. Baligar
e-mail: vpbaligar@kletech.ac.in

S. Fong et al. (eds.), *ICT Analysis and Applications*, Lecture Notes in Networks and Systems 154, https://doi.org/10.1007/978-981-15-8354-4_37

Compression is possible because of redundancy that exists in the digital image storage data. It may be in the form of coding, inter-pixel, and psycho-visual redundancy. By using variable length coding and some transform techniques, we can reduce these redundancies [1, 2].

In lossless compression, it is very much essential that the image has to be reconstructed accurately to achieve high quality. Lossless approaches use techniques like Huffman coding, LZW coding, run length coding, and arithmetic coding. [3]

Here, we present a lossy approach for image compression over grayscale images. Section 2 is literature survey, which provides an overview about some image compression techniques. In Sect. 3, we discuss about our proposed algorithm. Section 4 explains the compression and decompression algorithms. In Sect. 5, we discuss about 'correctness ratio,' and in Sect. 6, we provide experimental results, and Sect. 7 gives a comparison between our algorithm and the JPEG algorithm. Section 8 gives the conclusion.

2 Review of Literature

In this section, we discuss about some of the lossy and lossless image compression techniques and their outcomes.

As per Weinberger et al., 2000, low complexity lossless compression for images (LOCO-I) is the algorithm at the core of the new ISO/ITU standard, JPEG-LS for continuous-tone image compression. This algorithm gives good compression ratio, and level of complexity is also less [5].

Raid et al. discuss about lossy image compression algorithm using DCT which is used for full-color still image applications [6].

Baligar et al., 2006, discuss about the image coding algorithm based on fixed threshold method. Threshold is the peak absolute error (PAE) allowed in the decompressed image. Here, a comparison is made with SPIHT algorithm to show that this algorithm gives visually better images and execution time is less [7].

As per Sinisa ILIC, Mile PETROVIC, Branimir JAKSIC, Petar SPALEVIC, at lower values of bit rate, there arises noise effects from the compression methodology used in JPEG. Here contour-like structures appear, which are uncomfortable for better visibility [8].

Patil et al., discuss about a lossy compression algorithm using surrounding pixels method. Here, it is shown that at low PSNR levels, the number of exact pixels in reconstructed image increases, thus reducing the contour effects that may arise in JPEG at same PSNR values [9].

As per [4, 10–12], lossless techniques are discussed.

From the literature review, we understand that standard JPEG has some adverse effects at low PSNR values. Hence, we propose a pixel count approach by which quality can be improved at low PSNR values.

3 Pixel Count Approach Using Threshold Method

In this section, we present a pixel count approach using threshold algorithm. A grayscale image is processed in raster scan manner. Here, number of pixels is computed whose sum is lesser than a threshold, and this number is stored in a file instead of storing each pixel intensity value. Later, reconstruction is done by calculating the average of this count in a unique way, and each pixel value is thus reconstructed.

Here, threshold is taken as 255, and sum is computed as

$$\text{sum} = f[x, y] + f[x, y + 1] + \ldots <= 255$$

Later, difference is calculated as

$$\text{Difference} = 255 - \text{sum}$$

This count is saved in a file, and difference is saved in another file.
Reconstruction is done as shown below.
An average value is computed using

$$\text{Average} = \frac{255 - \text{difference}}{\text{Count}}$$

The number of pixels (=count) is replaced by this average value which is near to the actual value.

4 Methodology

Here, we describe the algorithms with examples taken over a sample Lena image. For our work, we have used the standard set of grayscale images of size 512×512.

4.1 Algorithm Used for Compression

1. Input the grayscale image pixel intensity values.
2. In the raster scan manner, count the number of pixels whose sum <=255

$$\text{Sum} = f[x, y] + f\,[x, y + 1] + \ldots f[x, y + n] \tag{1}$$

Condition is checked using Eq. (2)

$$f[x, y] + f[x, y+1] + \ldots f[x, y+n] \,(\text{count of pixels}) <= 255(\text{threshold}) \tag{2}$$

3. Store this count in a 'count' file.
4. Find the difference using Eq. (3)

$$\text{Diff} = \text{sum} - 255 \tag{3}$$

5. Store this difference in 'difference' file
6. Huffman encode the files.

Outputs generated are count value file and difference file.

4.2 Algorithm Used for Decompression

We can reconstruct the image using following algorithm using 'count' and 'difference' as input files

1. Input the values from count and difference files generated by compression.
2. Do the Huffman decoding.
3. Declare an array for image reconstruction.
4. C = count value, D = difference value
5. Diff = 255 − D
6. Reconstruct 'C' number of pixels with a value 'Avg'
   where Avg = Diff/C

Examples
We have applied the pixel count approach using threshold over grayscale images. Here, we discuss this algorithm with examples of two sample sets of Lena image.

Compression:
The input image is scanned in raster scan manner. The pixel intensity values of adjacent pixels are added till it is lesser than a threshold. In this case, threshold is taken as 255. The count value is stored in a file say 'count.' The difference between the sum and threshold is calculated and stored in a file say 'difference.' Figure 1a, b show two sample sets of Lena image.

Calculations for sample set 1:

1. Initially, first pixel which is 162 < 255 is considered (Fig. 1a). Since it is lesser than the threshold, i.e., 255, next pixel value is added which is again 162.
   162 + 162 = 324 which is greater than threshold, 255. Hence, count is stopped at 1 and the count is saved as '1' in 'count' file.
2. Next, compute the difference as, Difference = 255 − 162 = 93
   Ninety-three is stored in 'difference' file.

162	162	162	161	162	157
163	164	164	157	158	161
159	159	160	160	158	155

(a)

106	110	108	111	112	108
101	104	104	107	113	111
102	99	105	108	114	114

(b)

Fig. 1 **a** Sample set 1 of Lena image. **b** Sample set 2 of Lena image

Calculations for sample set 2:

1. In second sample (Fig. 1b), first pixel value is 106 and seccond pixel value is 110.
   $106 + 110 = 216$. If we add next pixel value, i.e., 108, $216 + 108 = 324$ which is greater than threshold. Hence, count is '2', and this '2' is saved in 'count' file.
2. Now compute the difference as $255 - 216 = 39$ and '39' is saved in 'difference' file.

Similarly, count and difference are computed for next pixels and stored in 'count' file as shown in Fig. 2a, b and 'difference' file as shown in Fig. 3a, b.

Decompression

Calculations for reconstruction:

Initially, encoded 'count' and 'difference' files are read and Huffman decoded. A two-dimensional array arr$[i, j]$ is declared to store reconstructed values.

A 'count' value is read as 'C', and 'difference' value is read as 'D'.

1. For sample set 1, first values from the two files are $C = 1$ and $D = 93$

1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1

(a)

2	2	2
2	2	2
2	2	2

(b)

Fig. 2 **a** Count file for sample set 1. **b** Count file for sample set 2

93	93	93	94	93	98
92	91	91	98	97	94
96	96	95	95	97	100

(a)

39	36	35
50	44	31
54	42	27

(b)

Fig. 3 **a** Difference file for sample set 1. **b** Difference file for sample set 2

Reconstruction is done using the equation, Avg = (255 − *D*)/*C*, Avg = (255 − 93)/1 = 162. Here, the count value '*C*' is 1. Hence, Avg value is assigned to first pixel which is exactly same as original, i.e., 162.

2. For sample set 2, *C* = 2 and Avg = (255 − *D*)/*C*, Avg = (255−39)/2 = 108.

Here, first two-pixel values are assigned the value 108, instead of 106 and 110 as 'Avg' value is assigned to '*C*' number of pixels where it results in slight loss.

Though it is lossy, reconstructed image quality is good which is measured by a metric called 'correctness ratio,' which is explained in Sect. 5. For the sample set 1, reconstruction is exactly same as original, but for sample set 2, there is slight loss. By this algorithm, we get exact reconstruction wherever the pixel intensity values are higher and results in no compression. But, we get more count wherever the image is darker and results in more compression ratio. (This is clear by the size of count and difference files for sample set 2).

5 Correctness Ratio—A New Quality Metric

The quality of reconstructed images can be assessed using metrics MSE and PSNR. But, here we propose a metric called correctness ratio (Co.R.) which gives a better method to test the quality of reconstructed images [8].

$$\text{Co.R.} = \frac{\text{Total of actual pixels in reconstructed image as that of original}}{\text{Total pixels in the original image}}$$

This threshold approach gives more accuracy as compared with JPEG, at same PSNR values, which is shown in Sect. 7. The number of actual pixels in the reconstructed image increases, thus increasing the correctness ratio. As this ratio increases, contours which may appear after reconstruction are reduced.

6 Results

We got the following results when we applied the proposed algorithm on standard set of images as shown in Fig. 4. Here, the reconstructed images show the quality of the images which seem to be near to original. Left side is the original images, and right side is the reconstructed images.

Fig. 4 Standard set of images with reconstructed images

Table 1 Comparison of number of correct pixels as that of original

Input files	Increased number of pixels as per proposed algorithm, compared to JPEG	Co.R. of JPEG	Co.R. of proposed approach
Lena	49,775	0.18	0.32
Baboon	72,867	0.05	0.33
Barbara	37,681	0.05	0.19
Airplane	75,764	0.1	0.38
Aya_matsuura	54,697	0.09	0.3
Pepper	45,656	0.08	0.23

7 Comparison Results

This section gives a comparison between the threshold algorithm with that of JPEG lossy. We compute the number of pixels in the reconstructed image, which are exactly same as that of original image using both algorithms, and we found that it is increased with proposed approach as shown in Table 1. Similarly, we compute correctness ratio for both algorithms, and it is found to increase as shown in Table 2.

8 Conclusion

The pixel count approach using threshold algorithm is an innovative and low computation-intensive method for image compression. The performance metric used, 'correctness ratio,' gives a count of, how many pixels have the same values as that of original. By this, we find that there is an increase in this number as compared to JPEG lossy, at same PSNR values. This shows that the contours which were appearing with JPEG algorithm can be reduced. Hence, we can say that the quality of reconstructed

Table 2 Comparison of correctness ratio (Co.R.)

Input files	PSNR in dB	Total pixels in the original image	Correct pixel count using JPEG approach	Correct pixel count using proposed approach
Lena	34.15	262,144	34,454	84,229
Baboon	31.2	262,144	14,322	87,189
Barbara	31.69	262,144	13,470	51,151
Airplane	34.5	262,144	26,336	102,100
Aya_matsuura	32.5	262,144	24,132	78,829
pepper	31.1	262,144	16,580	62,236

image is improved with this approach and compression ratio achieved is around three for standard data sets. Further, it can be tested for different threshold values and adaptive techniques for improvement.

References

1. Kaur, R., & Choudhary, P. (2016). A review of image compression techniques. *International Journal of Computer Applications*, *142*(1) (0975-8887).
2. Anju, & Ahlawat, A. (2016). Performance analysis of image compression technique. *International Journal of Recent Research Aspects*, *3*(2). ISSN 2349-7688.
3. Gonzalez, R. C., & Woods, R. E. (1978). *Digital image processing* (2nd ed.). Pearson Prentice.
4. Baligar, V. P., Patnaik, L. M., Nagabhushan, G. R. (2003). High compression and low order linear predictor for lossless coding of grayscale images. *Image & Vision Computing*, *21*, 543–550. www.elsevier.com.
5. Weinberger, M. J., Seroussi, G., & Sapiro, G. (1996). LOCO-I: A low complexity, context-based, lossless image compression algorithm. In *Proceedings of 1996 Data Compression Conference*, Snowbird, UT, Mar. 1996 (pp. 140–149).
6. Raid, A. M., Khedr, W. M., El-dosuky, M. A. & Ahmed, W. (2014). JPEG image compression using discrete cosine transform-A survey. *International Journal of Computer Science & Engineering Survey (IJCSES)*, *5*(2).
7. Baligar, V.P., Patnaik, L.M., Nagabhushan, G.R. (2006). Low complexity and high fidelity image compression using fixed threshold method. *Information Sciences, 176*, 664–675.
8. Ilic, S., Petrovic, M., Jaksic, B., Spalevic, P., Lazic, L., Milosevic, M. (2013). Experimental analysis of picture quality after compression by different methods. *Przegląd Elektrotechniczny*. ISSN 0033-2097, R. 89 NR 11/2013.
9. Patil, T. R., Baligar, V. P., & Huilgol, R. P. (2018). Low PSNR high fidelity image compression using surrounding pixels. In *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*, Kottayam, India (pp. 1–6). https://doi.org/10.1109/iccsdet.2018.8821082.
10. Novikov, D., Egorov, N., & Gilmutdinov, M. (2016). Local-adaptive blocks-based predictor for lossless image compression. In *2016 XV International Symposium "Problems of Redundancy in Information & Control Systems"*.

11. Huilgol, R. P., Baligar, V. P., & Patil, T. R. (2018). Lossless image compression using seed number and JPEG-LS prediction technique. In *Conference proceedings—Punecon 2018*.
12. Huilgol, R. P., Baligar, V. P., & Patil, T. R. (2018). Lossless image compression using proposed equations and JPEG-LS prediction technique. In *2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET)*, Kottayam, India (pp. 1–6). https://doi.org/10.1109/iccsdet.2018.8821065.