Reduction of Color Images using Averaging Functions

R. Afzal[1], A. K. Khan[2], G. Raja[3]
Department of Electrical Engineering, University of Engineering and Technology Taxila, Pakistan
[1]romana.afzal@gmail.com
[2]ahmad.khalil@uettaxila.edu.pk, [3]gulistan.raja@uettaxila.edu.pk

Abstract-In this paper, a color image reduction algorithm is implemented that makes use of averaging functions to reduce the images without decomposing them to RGB channels. Penalty functions are used to find out the most optimal averaging function that provides least error between original and reconstructed image. Simulation results show that the implemented algorithm reduces the size of the color images so that can be transmitted at low cost. The average values of MSE, MAE and SSIM in case of reduction of the 12 test images used our simulations are 250.73, 8.47 and 0.7217 respectively. Moreover, this scheme also does well when the test image is corrupted with noise.

Keywords-Averaging Functions, Image Reduction, Penalty Functions, Noise Reduction, Subsampling.

I. INTRODUCTION

Image reduction is an area of considerable importance in the field of image processing. Reduction is the process of changing the resolution or dimensions of the image to make it smaller in size, while keeping as much information as possible. Image reduction can be used to accelerate computations on an image or just to reduce the cost of its storage or transmission. It is a frequently performed operation in computer graphics, multimedia and electronic publishing [1].

Image reduction can be carried out by many methods. These methods can be divided in two groups. In the first group, the image is divided into blocks. The reduced image is made by combining the results of the algorithm in each block. For the second group, the image is considered in a global way [2], [3].

It means that it is treated as a whole. In the scheme implemented in this paper, the first group of algorithms (i.e. local algorithms) has been focused. Simple reduction algorithms can be designed working with small pieces of the image. When the reduced images are reconstructed, there exists some error between reconstructed and original image. The implemented scheme will provide image reduction using averaging functions such that the error between original and reconstructed images is minimum.

II. IMAGE REDUCTION TECHNIQUES

The most basic image reduction techniques are provided in this section. Their brief description is given as under:

A. Cropping

Cropping is the simplest way by which the size of an image can be reduced. It is useful only where a certain portion of an image is important enough to use. The rest of the image is discarded. The data loss caused by cropping cannot be retrieved. But uncropping is possible in two cases:

- Original copy of the image is safe
- Undo information is not deleted

B. Scaling

Scaling is also termed as Resizing. It is mainly concerned with number of pixels per inch (ppi) when the image is printed on paper. Scaling does not change the resolution of the image. It only shrinks or expands the pixels. So the image is not actually small rather it appears to be small. If the image is scaled to be enlarged, the pixels become increasingly visible, making the image appear soft if pixels are averaged or jagged if not averaged [4], [5].

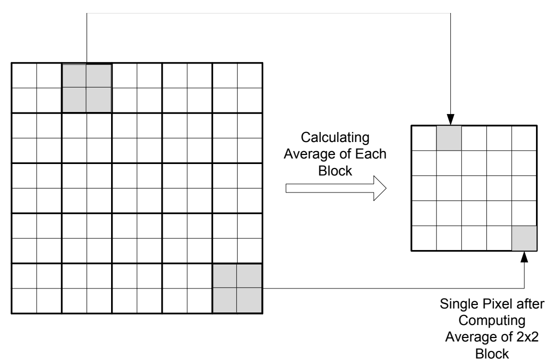


Fig. 1. Simple Image Reduction using Averaging Functions

C. Pixel Skipping/Subsampling

Pixel skipping or subsampling can reduce the size of an image by deleting pixels evenly throughout an image. For example, if an image is to be reduced by half in each dimension, every other row and column of pixels in the image must be deleted. This method is fast but it produces artifacts especially in color images [6]. It produces checkerboard effect in images when a higher order of reduction is to be achieved.

D. Pixel Averaging

The averaging functions (i.e. arithmetic mean, geometric mean, etc.) can be used for image reduction in such a way that original image is divided into blocks and each block is replaced by its average. For example, if an image is divided into 2x2 blocks as shown in Fig. 1, then computing average of each block will reduce the size of original image to one half [7]. For 3x3 blocks, size of the image will be reduced to one third of the original image.

III. AVERAGING FUNCTIONS

There are several types of averaging functions that are used in image processing. A few of these functions are:

A. Arithmetic Mean

The arithmetic mean is commonly known as *mean* or *average*. For a set of samples $\{x_i\}$, the arithmetic mean $\bar{x}$is calculated as:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad (1)$$

Arithmetic mean smoothes local variations in the images. It also reduces the noise by blurring the image.

B. Geometric Mean

The geometric mean of a sequence $\{a_i\}_{i=1}^{n}$ is given as:

$$G(a_1, \ldots, a_n) = \left(\prod_{i=1}^{n} a_i\right)^{1/n} \qquad (2)$$

It means that,

$$G(a_1, a_2) = \sqrt{a_1 a_2} \qquad (3)$$

$$G(a_1, a_2, a_3) = \sqrt[3]{a_1 a_2 a_3} \qquad (4)$$

and so on. Geometric mean achieves more smoothing effect in images as compared to the arithmetic mean, but it tends to lose image detail in the process.

C. Harmonic Mean

The harmonic mean $H(x_1, \ldots, x_n)$ of n numbers x_i (where i=1,....., n) is the number H defined by:

$$\frac{1}{H} = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{x_i} \qquad (5)$$

The special cases of n=2 and n=3 are given as:

$$H(x_1, x_2) = \frac{2x_1x_2}{x_1 + x_2} \qquad (6)$$

$$H(x_1, x_2, x_3) = \frac{3x_1x_2x_3}{x_1x_2 + x_1x_3 + +x_2x_3} \qquad (7)$$

and so on. The harmonic mean provides good results for salt noise and Gaussian noise, but fails in case of pepper noise.

D. Maximum

The largest value of a set or function is termed as *maximum*. The maximum of a set of elements $A = \{a_i\}_{i=1}^{N}$ is denoted as $maxA$ or $max_i a_i$, and is equal to the last element of a sorted (ordered) version of A. For example, given the set {3,5,4,1}, the sorted version is {1,3,4,5}, so the maximum is 5.Maximum selects the brightest pixel in each block. So, as a result, the reduced image is brighter than the original image.

E. Minimum

The smallest value of a set or function is termed as *minimum*. The minimum of a set of elements $A = \{a_i\}_{i=1}^{N}$ is denoted as $minA$ or $min_i a_i$, and is equal to the first element of a sorted (ordered) version of A. For example, given the set {3,5,4,1}, the sorted version is {1,3,4,5}, so the minimum is 1. The maximum and minimum are the simplest order statistic functions. Minimum selects the darkest pixel in each block. So, as a result, the reduced image appears to be darker than the original image.

F. Median

Median is an order statistic function that gives the *middle* value $\tilde{x}$ of a sample. It means that the value $\tilde{x}$is such that an equal number of samples are less than and greater than the value (for an odd sample size), or the average of the two central values (for an even sample size). For example, for data set {1,2,3,4,5}, median is equal to 3. Similarly, for data set {1,2,3,4,5,6}, median is equal to 7/2=3.5. Median is frequently used in image processing because it provides reduction in certain types of noises. Median filters have been used in the rovers in Mars in both navigation and science tasks.

G. Mode

Mode is the most common value obtained in a set of observations. For example, for a data set {3, 7, 3, 9, 9, 3, 5, 1, 8, 5}, the mode is 3. Similarly, for a data set {2, 4, 9, 6, 4, 6, 6, 2, 8, 2}, there are two modes: 2 and 6. A data set with a single mode is said to be unimodal. A distribution with more than one mode is said to be bimodal, trimodal, etc.

IV. Methodology

In the color image reduction algorithm implemented here, a number of different averaging functions are used. A color image is divided to *n*x*n* blocks.

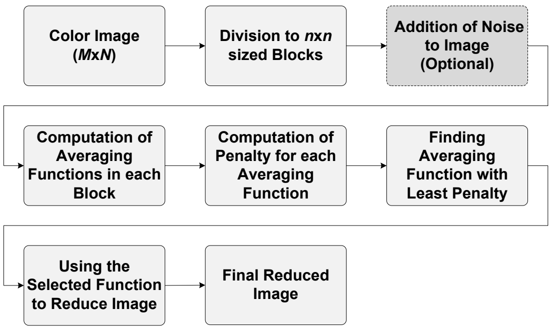


Fig. 2. Block Diagram of the Reduction Algorithm

The scheme mentioned in [7] splits the image to R, G, and B channels before applying the aggregation functions but this reduction algorithm does not do that. The averaging function that minimizes the penalty function is selected. Penalty function used in the reduction algorithm is given below:

$$P(x,y) = \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\left|x_{Cij} - y_C\right|\right)^2 \qquad (8)$$

This reduction algorithm does not have to decompose the image and then concatenate it at the end. Moreover, instead of 5 averaging functions (Arithmetic Mean, Geometric Mean, Maximum, Minimum and Median), 7 aggregation functions have been utilized including Harmonic Mean and Mode. A block diagram of implemented algorithm can be found in Fig. 2.

V. SIMULATION RESULTS

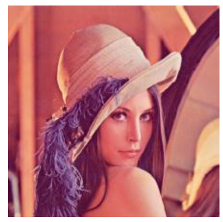
Twelve different color images (standard and common) were tested in MATLAB R2011b using reduction algorithm of color image reduction. All the test images are in TIFF and JPEG format and have different resolutions. Parameters of analysis are MSE, MAE and SSIM to compare original and reconstructed image [8]. The standard images used here can be found at http://www.imageprocessingplace.com/root_files V3/image_databases.htm.

A. Reduction Algorithm

Fig. 3 shows test color images that are to be reduced. The images have different resolutions but they are scaled such that they *appear* almost equal in size. Fig. 4 gives reduced images and Fig. 5 gives reconstructed images. Table I provides MSE, MAE and SSIM [9] to check the quality of reconstructed images.

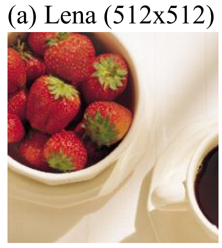
B. Response to Noise

This reduction algorithm responds greatly if the original image has been corrupted by salt and pepper or speckle noise. The averaging functions, especially arithmetic mean, geometric mean and median are known for removing the noise from images leaving them with a blurring and smoothing effect [10].



(a) Lena (512x512)

(b) Madril (512x512)



(c) Strawberries (664x664)

(d) Peppers (384x512)

(e) Library (360x480)

(f) Flowers (312x420)

(g) Chalk (1040x1040)

(h) Fruit (576x768)

(i) Balloons (548x548)

(j) Candy (852x1280)

(k) Grapes (576x768) (l) Berries (1068x1600)

Fig.3. Original Test Images

Fig. 6(a) shows the image corrupted with salt & pepper noise. Fig. 6(b) and Fig. 6(c) show the reduced and reconstructed images using subsampling and implemented algorithm. Similarly, Fig. 6(d) shows the image corrupted with speckle noise. Fig. 6(e) and Fig. 6(f) show the reduced and reconstructed images using subsampling and implemented algorithm.

(a) Lena

(b) Madril

(c) Strawberries

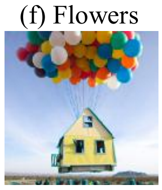
(d) Peppers

(e) Library

(f) Flowers

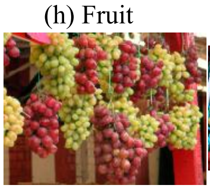
(g) Chalk

(h) Fruit



(i) Balloons

(j) Candy



(k) Grapes

(l) Berries

Fig. 4. Reduced Images (n=4)

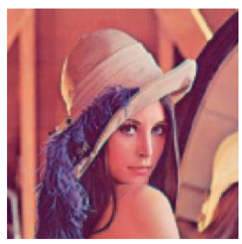
TABLE I

ERRORS AND STRUCTURAL SIMILARITY

Image	MSE	MAE	SSIM
Lena	131.61	6.55	0.7685
Mandril	623.45	17.87	0.4381
Strawberries	122.96	4.96	0.7976
Peppers	81.28	4.34	0.8606
Library	470.63	10.79	0.6632
Flowers	203.02	6.69	0.8296
Chalk	135.75	6.03	0.7592
Fruit	282.02	10.40	0.6661
Candy	359.84	12.62	0.5432
Balloons	162.09	4.95	0.8618
Grapes	206.51	8.34	0.7323
Berries	229.64	8.14	0.7409

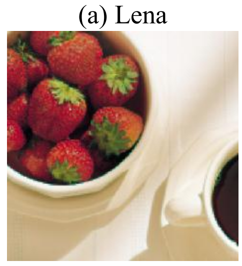
VI. CONCLUSIONS

The main purpose of this paper is to use the aggregation functions for color image reduction. In this context, an image reduction algorithm based on averaging and penalty functions has been implemented. This reduction algorithm is compared with the most commonly used image reduction method i.e. subsampling. It is found that reduction algorithm implemented in this paper is better for color image reduction. The implemented method efficiently filters out the speckle and salt & pepper noise from the images.



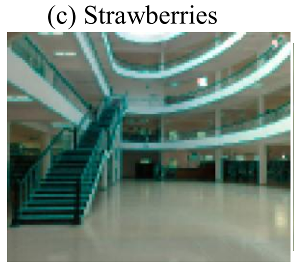
(a) Lena

(b) Madril



(c) Strawberries

(d) Peppers

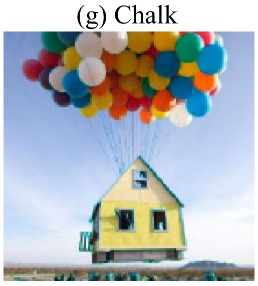


(e) Library

(f) Flowers

(g) Chalk

(h) Fruit



(i) Balloons

(j) Candy

(k) Grapes

(l) Berries

Fig. 5. Reconstructed Images

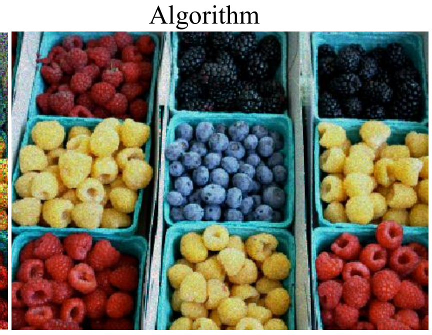
(a) Image with Salt & Pepper Noise

(b) Result of Subsampling

(c) Result of Implemented Algorithm

(d) Image with Speckle Noise

(e) Result of Subsampling



(f) Result of Implemented Algorithm

Fig.6. Response to Noise

References

[1] B. Shaick and L. Yaroslavsky. "Image Reduction for Object Recognition," *IEEE Region 8 International Symposium on Video/Image Processing and Multimedia Communications*, pp. 16-19, 2002.

[2] M. Trentacoste, R. Mantiuk and W. Heidrich. "Blur-Aware Image Downsampling," *The Eurographics Association and Blackwell Publishing Ltd*, vol. 30, 2010.

[3] D. Paternain, H. Bustince, J. Fernandez, G. Beliakov and R. Mesiar. "Image Reduction with Local Reduction Operators," *IEEE International Conference on Fuzzy Systems*, vol. 1, pp. 18-23, July 2010.

[4] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. USA: Prentice Hall, 2007.

[5] R. C. Gonzalez, R. E. Woods and S. L. Eddins. *Digital Image Processing using MATLAB*. USA: Pearson Prentice Hall, 2003.

[6] J. Sachs. *Image Resampling*. Massachusetts: Digital Light & Color, 2001.

[7] G. Beliakov, H. Bustince and D. Paternain. "Image Reduction Using Means on Discrete Product Lattices," *IEEE Transactions on Image Processing*, vol. 21, pp.1070-1083, 2012.

[8] K. N. Plataniotis and A. N. Venetsanopoulos. *Color Image Processing and Applications*. USA: CRC Press, 2000.

[9] L. Zhang, D. Zhang, M. Xuanqin and D. Zhang. "FSIM: A Feature Similarity Index for Image Quality Assessment,"*IEEE Transactions on Image Processing*, vol. 20, pp. 2378-2386, 2011.

[10] R. Samadani. "A Fast Algorithm for Preserving Noise while Reducing Image Size," *15th IEEE International Conference on Image Processing* , pp. 12-15, 2008.