BLIND IMAGE COMPRESSION HISTORY DETERMINATION USING DYNAMIC THRESHOLDING

Robert Grou-Szabo, Tadashi Shibata

Department of Frontier Informatics, Graduate School of Frontier Science,
The University of Tokyo, 5-1-5, Kashiwanoha, Kashiwa-shi, Chiba, 277-8561 Japan
grou@if.t.u-tokyo.ac.jp - shibata@ee.t.u-tokyo.ac.jp

ABSTRACT

In certain situations, a software application or even a hardware processing unit receives image data in a raster format, such as in a bitmap file, without any knowledge of what, if any, prior processing has occurred to it. If further processing is intended, it can be helpful or even necessary to know of an image's processing history. Using a directional edge detector with a dynamic thresholding technique, an algorithm as been developed that can determine an image's compression history using only the raster bitmap information. By accentuating an image's block boundaries and then subsequently comparing the edge count at the block boundary with the edge count the pixel row right below, the compression history of images compressed up to a Quality Factor of 100 can be determined.

Index Terms— video signal processing, image coding, noise filter

1. INTRODUCTION

There are several examples of both hardware and software applications where it can be useful to know the compression history (CH) of an image when all we have is a rasterized data format such bitmap or a data stream from an external RAM. For example in the case of rendering an image using a JPEG [10] compressed image for texture mapping that will be stretched or tiled or else when pasting a copied image directly into Word or PowerPoint from the Windows clipboard or when image data is to be converted to Postscript then sent to a printer driver, or simply when trying to recompress an image a second time. In each of these examples, block artifacts can be amplified, exaggerated or made to appear more obvious. However if it is known beforehand that the image has previously been compressed and contains block artifacts, measures can be taken and parameters changed to improve processing and possibly obtain better results.

Although our research borders on the "removal of block artifacts", for which numerous papers have been written, we are only interested in determining an image's Compression History Estimation (CHEst). Another of our main concerns is to develop a "hardware-friendly" approach while doing this, where the only information available is the decompressed bitmap information of a Monochrome image.

In addition, although there is no official standard value for the block size of a JPEG encoded image, we will assume that the grid is a regular pattern of square 8 by 8 pixel blocks. Thus, the only compression parameter to affect the image quality is the quantizer table. i.e. the quantizer step size applied to the DCT coefficients. The algorithm described in this paper does well even with images that have been compressed with a Quality Factor (QF) of 100 (where every DCT of the quantization table coefficient equals 1). Even when an image has been compressed with a QF = 100 does not mean that there is no loss. Even the lowest JPEG compression ratio (i.e. biggest file size), will still leave its mark on an image. So this algorithm can determine the CH at any compression ratio. Humans can identify objects from simple line images such as etchings or cartoons. Which leads us to believe that edge information is essential to the Human Visual System (HVS), even more so than color or light intensity is. So it stands to reason that edge detection is very important for image interpretation and identification. Therefore with this kind of reasoning we based our algorithm largely on edge detectors.

2. PREVIOUS WORK

Many papers have been put out proposing algorithms to remove JPEG block artifacts, most of which work using wavelets or Fourier transforms [2] or else need to use the DCT coefficients [4], [5]. However our research is different in that it focuses on detection using only uncompressed bitmap-like data. Moreover, our goal is to use the system in a hardware application, in that case an edge detector is much easier to implement and occupies less surface area than an FFT operator does. Similar research is being done by Fan and Queiroz in [1] where they identify an image's compression history from a bitmap image as well as estimate the quantizer table. However their algorithm uses block's difference of neighbors to detect a JPEG signature and although it is a simple and computationally fast algorithm, its very simplicity causes it to be sensitive to noise and distortion.

3. PROPERTIES OF JPEG ENCODING

The block artifacts or the blurring around sharp edges we see in reconstructed compressed images and video sequences are the result of coarse quantization and the truncation of high frequency coefficients. This is known as the Gibbs phenomenon and also appears as ringing noise or mosquito noise in images that have been encoded using a low bit rate. High frequency coefficients represent less visual information and can usually be discarded with little or no visible effect to the final image since a DCT operator essentially concentrates most of a signal's energy into just a few coefficients.

The JPEG group has come up with a Quality Factor (QF) in an attempt to quantify an image's quality after it has been compressed. A quality factor (QF) of 100 sets all the quantizer step sizes to unity and thus yields the best image quality JPEG can possibly achieve. It should be noted that even with the highest quality factor where each coefficient is '1', some information will be lost since the real DCT outputs are typically not integers. So although not visible, a JPEG signature still exists in a compressed image regardless of compression ratio.

4. COMPRESSION HISTORY DETERMINATION

4.1. Algorithm

References in this section refer to Figure 1. Once the image data is made available in a raster format, either as a bitmap file or read in from a RAM (a), the first step is to apply an edge detector to the image (b). Several different edge detector algorithms have been tested and we have found that simple directional edge detectors such as Prewitt or Sobel obtain better results as opposed to other more complex algorithms such as Canny, Zero-crossing or Marr-Hildreth, possessing extra steps such as contour following attempting to reduce the probability of false contours. Simple four directional edge detectors work best because basically we are trying to, statistically speaking; compare the number of edges at block boundaries with the number of edges one row above or below the block boundary.

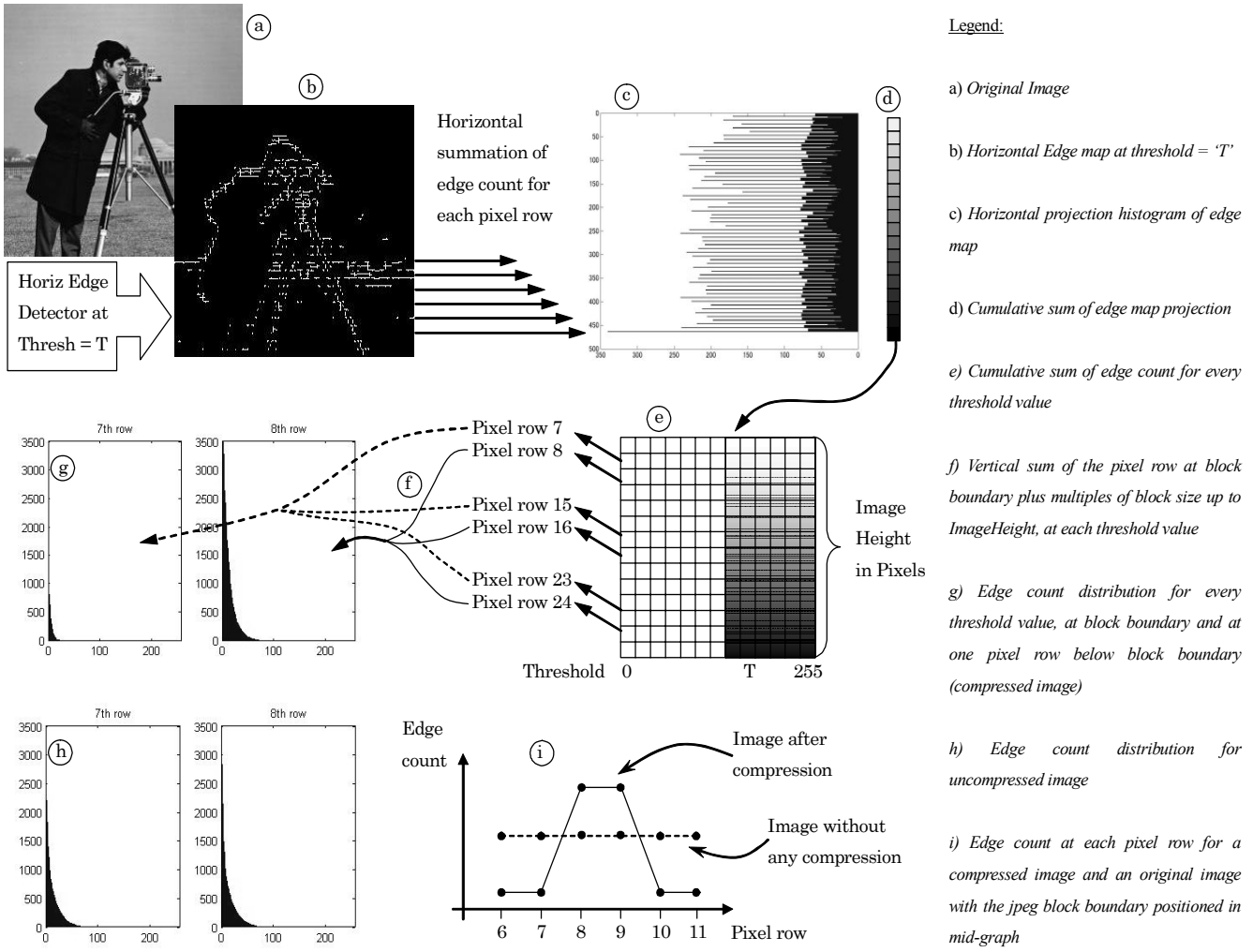

Fig. 1 Compression history determination algorithm

First the edge detector is applied (b) using a threshold value of zero '0'. At this point, essentially everything becomes an edge and the resulting edge map looks like Gaussian noise. Then we slowly increase the threshold value of the edge detector. As the threshold is increased, less and less edges appear in the edge map and a clearer image starts to appear. Thus the edge detector is passed over the image with a threshold value ranging from 0 the lowest numerical value for a pixel, to 255 the highest value.

For each threshold value, after the edge map is generated using a horizontal edge detector, a horizontal summation (c) of the edge map produces a vector that is as long as the image is high (d). One vector element for each pixel row of the image. Then, as every threshold value is used, from 0 to 255, the result is a matrix that is 256 x ImageHeight (e). This matrix contains all the edge information at every threshold value, for ever pixel row. The information that interests us are the rows that run along the JPEG block boundaries. Although there is no official standard value set down in the JPEG documentation, we have assumed that a block is 8 pixels high.

Next, we add together all the edge data for one threshold value, at every block boundary row. This means, for a threshold value of 'T', we sum together all the edge information for every multiple of the block size (f). More practically, what this means is that for one threshold value, we add together all the edge information for line 8 plus every multiple of block size, thus row 8 + row 16 + row 24 etc. The result of this summation is all the cumulated edge information at one particular threshold value of every multiple of the block boundary. The same is done for one row below the block boundary, i.e. row 7 + row 15 + row 23 etc. Effecting this summation for each threshold value, we obtain an edge map histogram (g) spanning from 0 to 255 representing all the edge information at the JPEG block boundary, as well as one pixel row below the block boundary.

By comparing the two edge map histograms from pixel row 7 and 8, we can observe that there is much more edge data on the block boundary when compared to one pixel row below it. The reason being, that JPEG block artifacts remain detectable by an edge detector much longer then edges contained in the scene, especially when cycling through the threshold values. We are using the statistical probability that JPEG edges, when looked at from a wide array of threshold values, will generate more edge data than regular edges in the scene.

We can see in (h) the result of an image that has not been JPEG encoded, keeping in mind that at this stage, the goal is to determine whether or not an image has been compressed. So if the result after passing an image though the algorithm looks like (g) then the image has at some time in the past been compressed and if the result looks like (h) then it's safe to assume that it has not.

4.2. Limitations

There are a few limitations to the algorithm. The first is that to be effective, the image has to be of a certain height. The height determines how many jpeg blocks an image has, and the more rows there are, the more statistical samples can be taken when using a horizontal edge detector. But this rule also applies to any statistical analysis where you need a big enough sample space to get an accurate study. Images smaller than 64 pixels in height tend to be blocky in appearance for that reason we have limited our test samples to images larger than 64 pixels high.

The second limitation is when the algorithm tries to analyze a computer generated image where an edge runs exactly along a pixel row so as to create a discrepancy in the edge count of one pixel row over another. This would be almost impossible to occur in an image taken from a real life situation. Any straight edge in an image would almost certainly cross over to one if not several other pixel rows.

5. EXPERIMENTAL RESULTS

Figure 2 shows the behavior of the edge count at each pixel row. In 2a, we see that there is no difference in edge count whether at the jpeg block boundaries or not, however 2b illustrates the behavior pattern that occurs once an image has been compressed and then assessed by the algorithm.

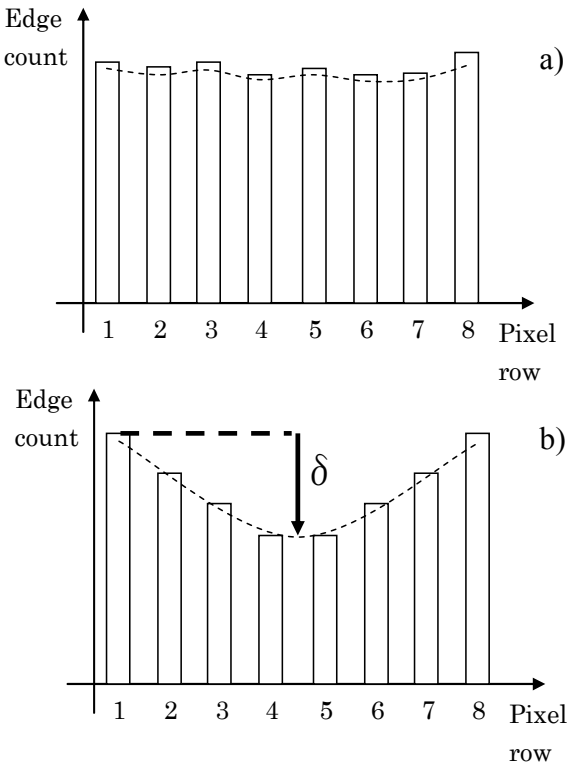

Fig. 2 Behavior of Edge count at each pixel row for a) uncompressed images and b) compressed image

The difference in edge count 'δ' can also give an idea of the compression ratio used when compressing the image. Images compressed at a high QF will still exhibit this kind of slope aspect but the difference in edge count will be much lower than images compressed with a low QF (i.e. small file size).

Figure 3 shows the result of the algorithm applied to 3 types of images; the four graphs show the resulting edge count when the images are compressed using different compression ratios. From left to right the JPEG compression ratios used are: QF of 100, 95, 85 and 75. The X axis represents the pixel rows 5, 6, 7... to 12 with pixel row 8 right at the mid point, the Y axis is the edge count for that pixel row. The images used were taken from various databases however special care was needed in certain cases since it might have been possible that an image having previously been passed from bitmap to JPEG and back to bitmap again would corrupt the results. Therefore images that were certain to have never been previously compressed at anytime were used. As well, the images used were all larger than 64 pixels high; most were at least 512 pixels high.

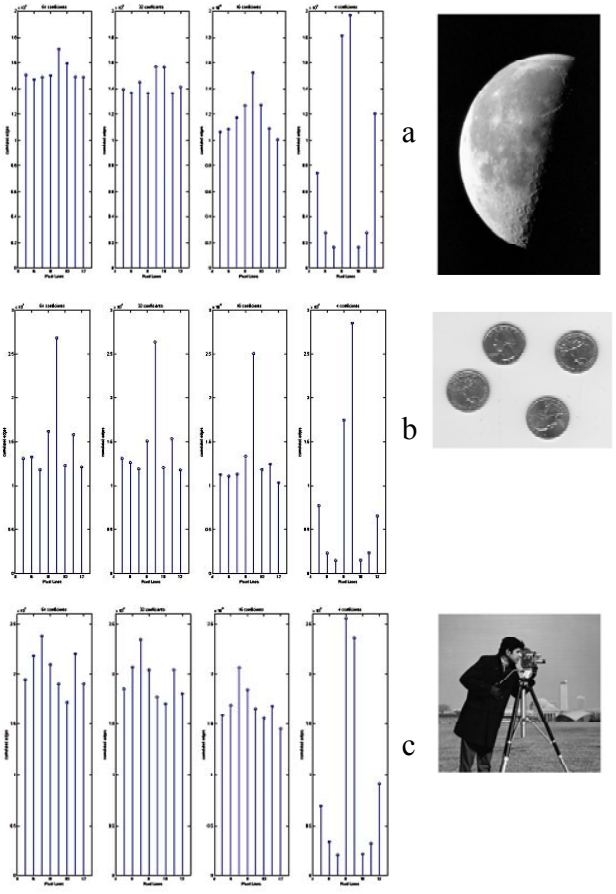

Fig. 3 Edge count of dynamic threshold edge detection

6. CONCLUSION

An algorithm to determine an image's compression history has been described. In some cases the maximum peak is located at the pixel row 8 which represents the bottom of the jpeg block boundary, however sometimes it was located at pixel row 9 which is the same as pixel row 1, i.e. the top of the jpeg block boundary. The confidence interval for this algorithm to determine any image's compression history mainly depends on two aspects of the image. The first is the image height which simply gives more or less statistical points for comparison.

The second is edge information density. A relatively smooth looking image with little or no edge information such as a sky or snow field will reveal more JPEG edge artifacts whereas an image loaded with edge information as in a crowded sports arena will confuse the algorithm and result in lower confidence interval.

7. REFERENCES

[1] Zigang Fan and Ricardo de Querioz, "Identification of Bitmap compression history: JPEG detection and quantizer estimation", IEEE Transaction on image processing, Vol. 12, No. 2, pp. 230-235, February 2003

[2] Ramesh Neelamani, Ricardo de Querioz, Zhigang Fan and Richard G. Baraniuk, "JPEG Compression history estimation for color images", IEEE Transaction on image processing, Vol. 15, No. 6, pp. 1365-1378, June 2006

[3] Hao-Song Kong, Anthony Vetro and Huifang Sun, "Edge map guided adaptive post-filter for blocking and ringing artifacts removal", in Proc. ISCAS, pp. 929-932, May 2004

[4] Shizhong Liu, Bovik, A.C., "Efficient DCT-domain blind measurement and reduction of blocking artifacts", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 12, No. 12, pp. 1139-1149, December 2002

[5] Irina Popovici and Wm. Douglas Withers, "Locating edges and removing ringing artifacts in JPEG images by frequency-domain analysis", IEEE Transaction on image processing, Vol. 16, No.5, pp. 1470-1474, May 2007

[6] Zhou Wang, Alan C. Bovik and Brian L. Evans, "Blind measurement of blocking artifacts in images", in Proc ICIP, pp. 981-984, September 2000

[7] Masakazu Yagi, Tadashi Shibata, "An Image Representation Algorithm Compatible to Neural-Associative-Processor-Based Hardware Recognition Systems," IEEE Trans. Neural Networks, Vol. 14, No. 5, pp. 1144-1161, September 2003

[8] Tadashi Shibata, "Intelligent Signal Processing Based on a Psychologically-Inspired VLSI Brain Model," IEICE Trans. Fundamentals, Vol. E85-A, No. 3, pp. 600-609 (2002)

[9] Zhou Wang and Alan C. Bovik, "A universal image quality index", in IEEE Signal processing letters, Vol. 9, No. 3, pp. 81-84, March 2002

[10] JPEG compression reference ONLINE