Computers and Electrical Engineering 46 (2015) 456–470

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

The pseudo-distance technique for parallel lossless compression of color-mapped images ☆

Basar Koc [a,*], Ziya Arnavut [b], Hüseyin Koçak [a]

[a] University of Miami, FL, United States
[b] SUNY Fredonia, NY, United States

ARTICLE INFO

Article history:
Received 15 May 2014
Received in revised form 7 January 2015
Accepted 9 January 2015
Available online 14 March 2015

Keywords:
Parallel image compression
Hyper-Threading
Color-mapped images
PDT
BWT
PNG

ABSTRACT

Data compression is a challenging process with important practical applications. Specialized techniques for lossy and lossless data compression have been the subject of numerous investigations during last several decades. Previously, we studied the use of the pseudo-distance technique (PDT) in lossless compression of color-mapped images and its parallel implementation. In this paper we present a new technique (PDT2) to improve compression gain of PDT. We also present a parallelized implementation of the new technique, which results in substantial gains in compression time while providing the desired compression efficiency. We demonstrate that on non-dithered images PDT2 outperforms PDT by 22.4% and PNG by 29.3%. On dithered images, PDT2 achieves compression gains of 7.1% over PDT and 23.8% over PNG. We also show that the parallel implementation of PDT2, while compromising compression less than 0.3%, achieves near linear speedup and utilization of Intel Hyper-Threading technology on supported systems improves speedup on average 18%.

© 2015 Elsevier Ltd. All rights reserved.

1. Introduction

For long term storage and efficient transmission of data, use of data compression is necessary. The purpose of a data compression process is to reduce the file size while minimizing the deterioration of information on the data. Reducing redundant data in a file is one of the common approaches used by most compression algorithms. Data compression consists of two main parts: encoding and decoding. Encoding is the process of generating data to represent original data in a more compact form. Decoding process is the reverse of the encoding process. In decoding, using compressed data, the original data is reconstructed.

There are two main types in data compression: lossy and lossless. In lossy compression, some information of the data may be ignored in encoding. When the encoded data is decoded, the original file and the decompressed file may differ. When portions of the data are not completely essential and minor loss in data is acceptable, lossy compression algorithms may be preferred in order to maximize compression gain. However, in some cases, e.g. medical images, when the quality of the image is top priority and any loss on data is not tolerable, the use of lossless compression algorithm is the preferred practice. In lossless compression, while the quality of a file is preserved, the reduction in the file size may not be as good as in lossy compression.

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Associate Editor Dr. Ferat Sahin.

* Corresponding author.

http://dx.doi.org/10.1016/j.compeleceng.2015.01.003
0045-7906/© 2015 Elsevier Ltd. All rights reserved.

Color image quantization [1,2] is a commonly used technique, especially for low-cost display devices. Such devices have certain limitations most notably the inability to display the full RGB color space. An image file that uses RGB color space may include 16.8 million (2^{24}) possible color combinations. To represent the whole spectrum, an RGB color image requires three bytes for each pixel; one byte for each color (red, blue, and green). The color quantization technique is applied to RGB images to map 24 bits to 16, 8, or 4 bits per pixel. In this technique, after creating a suitable palette to reduce number of colors, each pixel in the RGB image is replaced with the index of the closest color in the palette. When a palette size is 256 (2^8) or less, each index can be represented with 8 bits (a byte). Hence, the size of image can be reduced from three bytes to one byte per pixel by applying the quantization process.

During the quantization process, certain quantization errors, e.g. color distortion, may be introduced. To minimize quantization errors, utilization of dithering algorithms is an accepted practice. Many of the commonly used dithering algorithms employ some sort of a diffusion process where colors that are not available in the palette are approximated by a diffusion of colored pixels from within the available palette. Further information on dithering algorithms utilizing diffusion are available in, for example, [3,4].

Quantized images are stored and transmitted after they are compressed lossless with one of the standard image compressors, e.g. GIF, PNG, JPEG-LS, or JPEG2000. To improve compression gains, before using one of these compressors, re-indexing of the color table indices have been investigated by many researchers [5–8]. In the survey paper [5], Pinho and Neves compared various re-indexing schemes and concluded that "the pairwise merging heuristic proposed by Memon et al. is the most effective, but also the most computationally demanding." They also "found that the second most effective method is a modified version of Zeng's reordering technique, which was 3–5% worse than pairwise merging, but much faster." In a more recent publication [8], Battiato et al. proposed a new re-indexing algorithm and showed that "by using the PNG codec, the values of bpps of the proposed approach are considerably lower than the other methods; in some cases, the differences are substantial".

Instead of re-indexing color-table indices, Kuroki et al. in [9] proposed a different technique called *the pseudo-distance technique* (PDT) for lossless compression of non-dithered and dithered color-mapped images. PDT uses neighboring pixels to create a pseudo-distance table and transforms index values into pseudo-distance table values. In our earlier studies [10,11], we showed that better compression gains over other techniques are possible when PDT is applied to non-dithered color-mapped images along with an arithmetic coder. In [12], we examined the PDT for dithered color-mapped images and demonstrated that PDT is effective also on dithered images. Later, we investigated the use of a dynamic pseudo-distance table and in [13] showed that further improvement on compression gain is possible when the entries of the pseudo-distance table are updated based on the neighboring pixels.

The increased availability and lower prices of multi-core central processing units (CPU) for personal computers and mobile devices created an interest in parallelization of commonly used data compression algorithms. Howard and Vitter [14] proposed an efficient parallelization of Huffman and arithmetic coder for lossless image compression. Gilchrist [15] developed a parallel implementation of bzip2. In [16], Gilchrist and Cuhadar presented various parallel algorithms of block sorting technique, e.g., Burrows–Wheeler Transform (BWT) for lossless image compression. In [17], we showed that PDT is highly parallelizable and suitable for systems with multi-core CPUs. When we parallelized the PDT algorithm in [11], we obtained near linear speedup and showed that by using the parallel implementation of PDT, compression time can be reduced significantly for lossless compression of the non-dithered color-mapped images.

In this paper, first, we present a new pseudo-distance technique (PDT2) that improves the efficiency of our previous sequential PDT algorithm. The main improvements are: PDT uses three neighbors of a pixel to encode the pixel value but PDT2 uses up to five neighbors of the pixel; while the PDT algorithm updates one row of the pseudo-distance table for each pixel, PDT2 may update up to five rows based on the availability of neighbors of the pixel. In addition, PDT2 swaps the entries of a pseudo-distance values depending on neighboring pixels to better predict error values from the pseudo-distance table.

Second, we parallelize the PDT2 using the same technique that we utilized in [17] to parallelize PDT. Furthermore, we evaluate the performance of the parallel implementation of PDT2 along with the Hyper-Threading technology on the dithered and non-dithered color-mapped images. With PDT2, we improve the compression gain 22.4% over our previous study [11] on the non-dithered images, 7.13% on the dithered images [13]. The speedup with the parallelization of PDT2 agrees with the theoretical prediction form Amdahl's Law [18], with only 0.3% compression loss, and in case of Hyper-Threading with Casey's Law [19].

This paper is planned as follows: In Section 2, the description of the PDT2 is outlined. In Section 3, the description of the block-sorting transformation and the arithmetic coder used is presented. In Section 4, the experimental results are discussed, and finally, in Section 5, the conclusions of our paper are summarized.

2. PDT2

In this section, we briefly describe the Euclidean distance, encoding and decoding algorithms of PDT2, and a parallel implementation of PDT2 algorithm.

Download English Version:

https://daneshyari.com/en/article/455251

Download Persian Version:

https://daneshyari.com/article/455251

Daneshyari.com