Lossless Image Compression using Zipper Transformation

Babajide O. Ayinde\textsuperscript{1} and Ahmed H. Desoky\textsuperscript{2}
\textsuperscript{1}ECE Department, \textsuperscript{2}CECS Department, University of Louisville, Louisville, KY 40218. {babajide.ayinde, ahmed.desoky}@louisville.edu

Abstract—This paper proposes a lossless compression scheme for greyscale images using Zipper Transformation (ZT) and Inverse Zipper Transformation (iZT). The proposed transformation exploits the conjugate symmetry property of DFT. We benchmark the proposed ZT with both Discrete Cosine Transformation (DCT) and Fast Walsh Hadamard Transformation (FWHT) in order to quantify the efficacy of the proposed transformation. Numerical simulations show that ZT-based compression algorithm is lossless and gives a faster implementation than its counterparts. The experiments we performed for different block sizes also reveal that the ZT-based algorithm outperforms the FWHT-based algorithm in terms of how much they losslessly compress the original image.

1. Introduction

Lossless image compression has been found useful in many real world applications such as biomedical image analysis, art images, security and defense, remote sensing, just to mention a few [1]. In most medical applications, images are acquired and stored digitally. This is mostly true in radiology application where the images are grayscale. The images are always very large in size and number, and any means to losslessly compress them can lead to the reduction of the storage cost and improvement in the speed of transmission. Even though the cost of storage and transmission of digital signal has plummeted, however, the demand for lossless compression of medical images is exponentially increasing [2].

In a general sense, image compression can be divided into two main categories: lossy and lossless compression. Lossy compression deals with compression schemes that tolerate some certain amount of error, that is, the compressed and the decompressed images are not perfectly identical. In contrast, lossless compression encodes all the information from the original image and therefore, the decompressed image is exactly the same with the original image.

Lately, the sophistication and complexity of compression paradigms are also increasing with increasing speed of computing. It must be remarked that the cost of compression must be accounted for and considered accordingly. The cost of implementing and deploying a compression scheme is proportional to its complexity. One way to mitigate the complexity is by deploying proprietary methods. However, these methods are detrimental and come with a cost, and one of the common problem is inter-operability with existing equipments [3–5]. For the purpose of striking a balance between complexity and cost, a good number of recent proposed methods focused on lossy compression. In lossy compression, information that is not of significant importance is deliberately discarded. Better compression can be achieved if some visible losses can be tolerated for the clinical task purposes. In a sense, there are still a lot of controversies as to what the real life applications of lossy compressions are, especially in the medical field. One other approach to lossy compression is the machine learning approach where images are encoded with a sparse feature representations using autoencoders [6–9].

Generally speaking, lossless compression can be categorized into three broad categories, namely: predictive scheme with statistical modeling, transform based coding and finally, dictionary based coding. The predictive deals with using statistical method to evaluate the differences between pixels and their neighbors, and performed context model before coding. Whereas in transform based compression, pixel are transformed using frequency or wavelet transformation before modeling and coding. Dictionary based compression is the third category and it deals with replacing strings of symbols with shorter codes. It must be noted that dictionary based schemes are widely used for text compression [10]. An example of a dictionary based algorithm is the well known ZIP package. Other dictionary based compression algorithms for image data are the Lempel-Ziv-Welsh (LZW), Portable Network Graphics (PNG), Graphics Interchange Format, and so on.

Many decades ago, the JPEG image coding standard is two-fold: The first involves lossy image compression using Discrete Cosine Transform (DCT) and entropy; the second fold deals with lossless reconstruction of the original image using both the predictive scheme and entropy coding. Nowadays, lossless compression typically...
deals with both predictive and statistical modeling. One of the new lossless compression paradigms is the JPEG 2000 scheme which utilizes wavelet transformation. In this paper, we propose a new way to losslessly compress grayscale images and evaluate its performance in comparison with two existing lossless compression paradigms for grayscale images.

![Fig. 1: Types of Lossless Compression](image)

The rest of the paper is organized as follows. Section II gives the state of the art on lossless compression methods. Section III describes our implementation of the zipper, inverse zipper transformation, zipper-based Huffman coding and section IV discusses the experimental designs and presents the results. Finally, conclusions are drawn in section V.

### 2. Related work

A good number of image data compression paradigms have been investigated in the recent past. The least squares adaptive prediction scheme was proposed in [11] for lossless compression of natural images and the authors show that their novel scheme improves the computational complexity with negligible performance trade-off. Lossless compression that is based on adaptive spectral band re-ordering and adaptive backward previous closest neighbors algorithms (PCN) was also proposed in [12], [13] for hyperspectral images, and it was shown that the compression performance was greatly enhanced with the implementation of both the re-ordering of spectral band. The problem of compression in medical applications has also been dealt with. For instance in [14], different types of compression standards on grayscale medical images were compared by the authors, and the pros and cons of each of the methods were highlighted.

In [1], problems of video compression is addressed taking cognizance of the temporary spectral information. Also in [15], the possibility of using 3-D versions of the lossless JPEG spatial predictors was considered and the likelihood of using best predictor, determined on the basis of the previous frame, to encode the present frame was investigated.

The spectral redundancy was also exploited by implementing the best predictor from one spectral component to another spectral component. It can be inferred from the paper that pixels in a given neighborhood are concurrent in adjoining color bands. To improve on this, a different predictor for interband correlation was proposed in [16]. The authors in [17] also proposed a simple context predictive scheme where both intraframe or interframe coding is selected on the basis of temporal and spatial variations, and they then computed the prediction of the current pixel. A good number of predictors are considered taking cognizance of the spatial redundancy.

The Huffman coding technique is a commonly used scheme for data compression because it is very simple and effective. It requires the statistical information about the distribution of the data to be encoded. Besides, an identical coding table is used in both the encoder and the decoder. In [18], a new image lossless compression scheme based on Huffman coding was presented. The scheme is implemented in two stages - the linear predictor stage and the Huffman coding stage. It was shown by the authors that the propose scheme has the capability of reducing the Huffman coding table while improving the compression ratio. Improving the lossless compression of images with sparse histogram is the main concern in [19], and it was shown that the proposed scheme is also robust on other images. Both lossy and lossless compression was investigated in a unified framework and a new cost effective coding strategy was proposed to enhance the coding efficiency. A new motion-JPEG-LS based lossless scheme was also proposed in [2] and the authors only explored high enough correlation between adjacent image frames in order to avoid the possible coding loss and abrupt high computational cost. Also in [20], a wavelet-based lossy to lossless ECG compression was proposed. Lossless digital audio compression scheme was proposed and many other audio compression technologies were also mentioned in [21]. The main contribution of this paper is to propose a new lossless compression for grayscale images and benchmark its performance with DCT and FWHT compression paradigms for grayscale images in term of computational complexity, run time, and compression ratio.

### 3. Software Description

#### 3.1 Zipper Transformation

In this transformation, we exploited the conjugate symmetry property of DFT. The main idea in this transformation is to first carry out a Discrete fast fourier transform (DFT) on the input array. We then extracted...
The complex elements of the vector in the upper half of the symmetry. The imaginary part of the complex numbers in the upper half of the symmetry are stripped off and concatenated with their corresponding real counterpart. This procedure is reversed in the inverse zipper transformation. Fig 5 illustrates how the zipper and inverse zipper transformations are implemented using a $10 \times 1$ vector. It must be remarked that the number of elements in the vector before ZT stays the same after the transformation, that is, ZT preserves the dimensionality of the input data. It must also be noted that this transformation is absolutely lossless. Also, the two-dimensional zipper transformation is implemented by performing ZT on all the columns of the input matrix and then on all the rows.

### 3.2 Huffman Coding

This method was proposed in 1952 by David Huffman to compress data by reducing the amount of bits necessary to represent a string of symbols. Huffman coding use codewords with variable length, and with shortest codewords for most frequently used characters. The flowchart of the Huffman coding is given below:

![Huffman Coding Flowchart](image)

Greyscale image with no loss incurred. The schematic of the whole process is as shown in Fig 4.

### 4. Experimental Setup and Results

#### 4.1 Dataset

In the experiments, we used five grayscale images that are available online namely: lenna.jpg - size $512 \times 512$, elaine.gif - size $512 \times 512$, cameraman.png - size $256 \times 256$, man.tif - size $512 \times 512$, and couple.png - size $512 \times 512$. These are visualized in Fig 5.

#### 4.2 Experimental Design

We carried out the experiments in MATLAB environment and we report the standard metrics: the compression ratio (CR) and the running time in (seconds) evaluated on a machine with Intel(r) Core(TM) i7-6700.
Input random Variables

![Diagram](attachment:image.png)

**Fig. 4:** Lossless image compression pipeline using Zipper Transformation

CPU @ 3.40Ghz and a 64GB of RAM running a 64-bit Windows 10 Enterprise edition. The compression ratio (CR) is an important criterion in choosing a compression scheme for lossless image compression. The criterion is used to compare different compression paradigms, and is defined as:

\[
CR = \frac{\text{Original file size}}{\text{Compressed file size}}
\]  

(1)

In order to benchmark the proposed scheme with other methods, we also implemented DCT and FWHT. We also utilized the running time to compare the proposed method with DCT and FWHT based compression algorithm. In this work, we define the running time as the time elapsed between the zipper transformation and the inverse transformation as shown in Fig. 4. The MATLAB implementation of these algorithms can be downloaded from https://github.com/babajide07/Zipper-Transformation.

### 4.3 Performance Evaluation

As shown in Fig 6-10, the ZT-based compression algorithm outperforms both the DCT and FWHT counterparts for different block sizes. This performance is more pronounced when the block size is 64. In addition to the poor compressing capability, implementing DCT and FWHT are more expensive in terms of running time.

#### Table 1: Comparison table for DCT, FWHT and ZT using Lenna Image

<table>
<thead>
<tr>
<th>Block Size</th>
<th>DCT Entropy</th>
<th>DCT Lenght</th>
<th>FWHT Entropy</th>
<th>FWHT Lenght</th>
<th>ZT Entropy</th>
<th>ZT Lenght</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2.746</td>
<td>2.678</td>
<td>2.957</td>
<td>3.148</td>
<td>2.822</td>
<td>2.778</td>
</tr>
<tr>
<td>8</td>
<td>1.427</td>
<td>1.724</td>
<td>2.072</td>
<td>2.039</td>
<td>1.617</td>
<td>1.839</td>
</tr>
<tr>
<td>16</td>
<td>0.983</td>
<td>1.465</td>
<td>1.716</td>
<td>1.606</td>
<td>1.025</td>
<td>1.294</td>
</tr>
<tr>
<td>32</td>
<td>0.792</td>
<td>1.217</td>
<td>0.846</td>
<td>1.320</td>
<td>0.516</td>
<td>1.195</td>
</tr>
<tr>
<td>64</td>
<td>0.415</td>
<td>1.110</td>
<td>1.006</td>
<td>1.167</td>
<td>0.873</td>
<td>1.071</td>
</tr>
<tr>
<td>128</td>
<td>0.386</td>
<td>1.069</td>
<td>0.648</td>
<td>1.083</td>
<td>0.217</td>
<td>1.053</td>
</tr>
</tbody>
</table>

#### Table 2: Comparison table for DCT, FWHT and ZT using Elaine.GIF Image

<table>
<thead>
<tr>
<th>Block Size</th>
<th>DCT Entropy</th>
<th>DCT Lenght</th>
<th>FWHT Entropy</th>
<th>FWHT Lenght</th>
<th>ZT Entropy</th>
<th>ZT Lenght</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3.2716</td>
<td>2.3286</td>
<td>2.4835</td>
<td>2.4109</td>
<td>2.3783</td>
<td>3.1699</td>
</tr>
<tr>
<td>8</td>
<td>2.0114</td>
<td>2.1092</td>
<td>2.1811</td>
<td>2.2509</td>
<td>1.8831</td>
<td>1.9409</td>
</tr>
<tr>
<td>16</td>
<td>1.0158</td>
<td>1.4790</td>
<td>1.4543</td>
<td>1.4775</td>
<td>1.1149</td>
<td>1.3087</td>
</tr>
<tr>
<td>32</td>
<td>0.5835</td>
<td>1.1591</td>
<td>1.2995</td>
<td>1.3012</td>
<td>1.1911</td>
<td>1.4691</td>
</tr>
<tr>
<td>64</td>
<td>0.2771</td>
<td>1.1365</td>
<td>0.6823</td>
<td>1.1042</td>
<td>0.4188</td>
<td>1.0507</td>
</tr>
<tr>
<td>128</td>
<td>0.1540</td>
<td>1.0378</td>
<td>0.2200</td>
<td>1.0772</td>
<td>0.1008</td>
<td>1.0216</td>
</tr>
</tbody>
</table>

#### Table 3: Comparison table for DCT, FWHT and ZT using Cameraman.PNG Image

<table>
<thead>
<tr>
<th>Block Size</th>
<th>DCT Entropy</th>
<th>DCT Lenght</th>
<th>FWHT Entropy</th>
<th>FWHT Lenght</th>
<th>ZT Entropy</th>
<th>ZT Lenght</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>3.3260</td>
<td>3.1843</td>
<td>3.3128</td>
<td>3.3539</td>
<td>2.9919</td>
<td>3.0521</td>
</tr>
<tr>
<td>8</td>
<td>2.5294</td>
<td>2.3920</td>
<td>2.6009</td>
<td>2.4979</td>
<td>2.0671</td>
<td>2.1729</td>
</tr>
<tr>
<td>16</td>
<td>1.7815</td>
<td>1.8506</td>
<td>2.0891</td>
<td>2.0168</td>
<td>1.4019</td>
<td>1.9002</td>
</tr>
<tr>
<td>32</td>
<td>1.3217</td>
<td>1.4974</td>
<td>1.5239</td>
<td>1.7676</td>
<td>1.1557</td>
<td>1.4082</td>
</tr>
<tr>
<td>64</td>
<td>0.8406</td>
<td>1.2657</td>
<td>1.0793</td>
<td>1.5098</td>
<td>0.4568</td>
<td>1.1487</td>
</tr>
<tr>
<td>128</td>
<td>0.5813</td>
<td>1.1440</td>
<td>0.5756</td>
<td>1.1645</td>
<td>0.5211</td>
<td>1.0637</td>
</tr>
</tbody>
</table>
than the ZT-based lossless compression. For instance in Fig 6, with block sizes of 4 and 8, DCT and zipper transform have similar running time, however as the block size increases, zipper transform performs better than DCT and FWHT in terms of both the compression and running time. Tables I-V give the average length of the codewords and the entropy for all the three methods that were compared in this study. It can be observed that the average length of the codeword decreases as the entropy decreases for all the three methods, and also the value of entropy is very close to the average codeword length.
Fig. 8: A plot of (a) compression ratio (b) running time of DCT, FWHT and Zipper against the block size using cameraman.png.

Fig. 9: A plot of (a) compression ratio (b) running time of DCT, FWHT and Zipper against the block size using man.tif.

Fig. 10: A plot of (a) compression ratio (b) running time of DCT, FWHT and Zipper against the block size using couple.png.
5. Conclusion

We show in this study that using zipper transform based algorithm, a better compression can be achieved. The proposed algorithm also performed better than both DCT and FWHT based algorithms in terms of how much they are able to compress the image, and also, the implementation time of zipper based compression is superior to the other two methods we considered.

References