Grayscale Image Compression Using Wavelets

MATH 719	Final Project
Submitted by:	Nathan King
Student #:	301222376
Submitted to:	Dr. Paul Tupper
Date:	July 31, 2014

Contents

1 Introduction **2**

2 Discrete Wavelet Transform **2**

2.1 Multiresolution Analysis 3

2.2 Two-Dimensional Wavelets 5

3 Wavelet Based Compression Algorithms **6**

3.1 Baseline Wavelet Compression Algorithm 7

3.2 Embedded Zerotree Wavelet Algorithm 8

4 Numerical Results **11**

5 Discussion **15**

References **16**

1 Introduction

Popular use of wavelets as a mathematical tool is quite recent. The modern wavelet transform was first developed in 1981 by Jean Morlet and Alex Grossman [1]. Since then a complete theory has been developed and many applications have been explored. Solving partial differential equations [2], analyzing Brownian motion [3], artist identification [4], are just a few applications that can involve wavelets.

The use of wavelets in image compression is investigated here. The purpose of image compression is to eliminate redundancy in image data to increase efficiency of data transfer (or to store data more compactly) [5]. There are two main types of image compression: *lossless* and *lossy.* Lossless image compression does not lose any information in the process of compression. This means that lossless compression has the ability to be decompressed perfectly (i.e. original image is recovered). Lossy image compression loses information during compression. Therefore the image can not be decompressed without differences from the original.

It is difficult to obtain an error free lossless compression with better than 2:1 compression ratio [6]. This paper therefore investigates types of lossy wavelet based image compression. The first is a simple baseline algorithm that helps with the understanding of wavelet based image compression [5]. The second is called the *embedded zerotree wavelet* (EZW) algorithm, which was introduced by Shapiro in [7]. The final method is the *set sartitioning in hierarchical trees* (SPIHT) algorithm [8].

The paper unfolds as follows. Section 2 details the discrete wavelet transform (DWT) and describes some common wavelets. Common steps of wavelet based image compression are discussed in section 3. It is also in section 3 that the baseline and EZW compression algorithms are explained. A comparison of all three methods, applied to grayscale images, is given in section 4. We conclude in section 5 with a discussion of the overall preformance of the methods and some information on other wavelet based compression algorithms.

2 Discrete Wavelet Transform

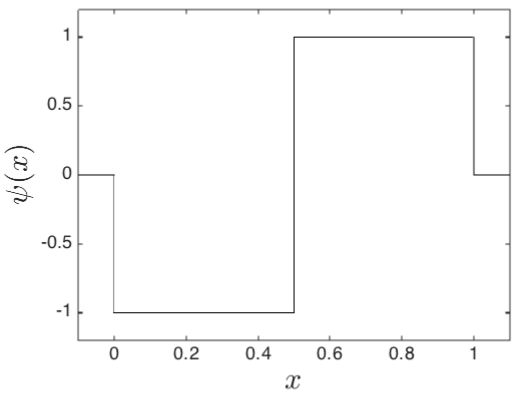
From a given image, the goal of the compression is to minimize the sequence of bits needed to represent it, while preserving information of acceptable quality. Wavelets address this problem by decomposing an image in a manner that acts like the human visual system [6]. The image is expressed in terms of its *average,* a_1 and successive levels of *detail,* $d_1, d_2, \ldots$. Similar to shooting on net in hockey; one first identifies the location of the net, then focuses in on the top corner to shoot the puck. This zooming mechanism is inherent in the multiresolution structure of wavelet bases.

The discussion of wavelets here is for the space $L^2(\mathbb{R})$ and follows that of Pereyra and Ward [1]. Wavelets are a basis for $L^2(\mathbb{R})$ that are generated from scalings and translations of one function $\psi(x)$.

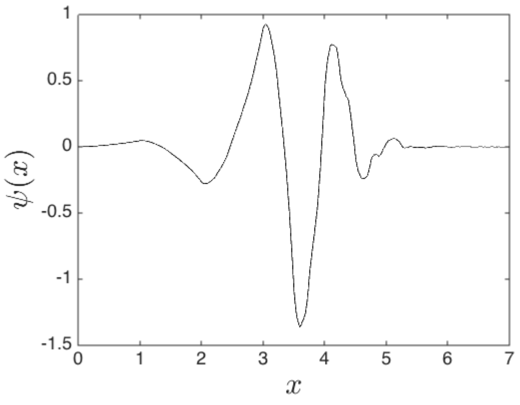
Definition 1. *A function $\psi \in L^2(\mathbb{R})$ is a wavelet if the family*

$$\psi_{j,k}(x) = 2^{j/2}\psi(2^j x - k) \quad \text{for } \; j,k \in \mathbb{Z} \tag{1}$$

forms an orthonormal basis of $L^2(\mathbb{R})$.



a) Haar mother wavelet



b) Daubechies 4 mother wavelet

Figure 1: Plots of two mother wavelets used in this paper. Haar mother wavelet is the discontinuous and compactly supported wavelet in a). Daubechies 4 mother wavelet, given in b), is continous and compactly supported.

The function $\psi(x)$ is a *mother* wavelet and $\{\psi_{j,k}\}_{j,k\in\mathbb{Z}}$ is called a *wavelet basis*. Examples of two different mother wavelets are given in figure 1. The Haar mother wavelet is given by $\psi(x) := -\chi_{[0,1/2)}(x) + \chi_{[1/2,1)}(x)$ and is shown in figure 1a. The Haar wavelet is the simpliest known wavelet. The Daubechies 4 mother wavelet is shown in figure 1b. These two wavelets will be used in the numerical experiments of section 4.

From the completeness of an orthonormal basis the reconstruction formula holds

$$f(x) = \sum_{j,k\in\mathbb{Z}} \langle f, \psi_{j,k}\rangle \; \psi_{j,k}(x) \quad \text{for all } \; f \in L^2(\mathbb{R}), \tag{2}$$

where equality is in the L^2 sense. The inner product, $\langle f, \psi_{j,k}\rangle$, defines the (j,k) wavelet coefficient. That is, the *orthonormal wavelet transform* is the map $W : L^2(\mathbb{R}) \to l^2(\mathbb{R})$ that assigns to each function in $L^2(\mathbb{R})$ the sequence of wavelet coefficients

$$Wf(j,k) := \langle f, \psi_{j,k}\rangle = \int_{\mathbb{R}} f(x)\overline{\psi_{j,k}(x)}\, dx. \tag{3}$$

The difficult part of wavelet theory is finding the wavelets themselves. The Haar mother wavelet on $L^2([0,1))$ is the earliest know wavelet introduced by Alfréd Haar in 1910 [9]. It is nice to have formulations (2) and (3) for reconstructing f and computing wavelet coefficients. However, the multiresolution structure of wavelets provides deeper understanding and introduces methods to construct more general wavelets.

2.1 Multiresolution Analysis

Mulitresolution Analysis (MRA) is an abstract formulation for the idea of writing functions in terms of dilates and translates of a wavelet function. Only orthogonal MRA are discussed here. Due to the abstract nature of MRA, it may be difficult at first to digest.

Definition 2. *An orthogonal MRA, with scaling function φ, is a collection of closed subspaces $\{V_j\}_{j\in\mathbb{Z}}$ of $L^2(\mathbb{R})$ such that*

1. $V_j \subset V_{j+1}$ *for all* $j \in \mathbb{Z}$,
2. $\bigcap_{j\in\mathbb{Z}} V_j = \{0\}$,
3. $\bigcup_{j\in\mathbb{Z}} V_j$ *is dense in* $L^2(\mathbb{R})$,
4. $f(x) \in V_j \Leftrightarrow f(2x) \in V_{j+1}$,
5. $\varphi \in V_0$, *and its integer translates,* $\{\varphi(x-k)\}_{k\in\mathbb{Z}}$, *form an orthonormal basis for* V_0.

Definition 2 says that an orthogonal MRA is a collection of closed subspaces of $L^2(\mathbb{R})$ that: 1) are nested, 2) have trivial intersection, 3) exhaust all of $L^2(\mathbb{R})$, 4) have subspaces that communicate through a scaling property and 5) the integer translates of φ form an orthonormal basis for the *central space* V_0. The subspaces $\{V_j\}_{j\in\mathbb{Z}}$ are called the *approximation* subspaces.

The scaling function φ belongs to the central space V_0. Since the spaces are nested $\varphi \in V_0 \subset V_1$, so φ also belongs to V_1. The scaled translates $\sqrt{2}\varphi(2x-k)$ of $\varphi(2x)$ form an orthogonal basis for V_1. Hence $\varphi(x)$ can be written in terms of these scaled translates

$$\varphi(x) = \sqrt{2}\sum_{k\in\mathbb{Z}} h_k \varphi(2x-k), \tag{4}$$

which is the *dilation equation*. The coefficients h_k are called the *filter coefficients* and fully describe an MRA. Two other natural ways of specifying an MRA are by the approximation subspaces V_j or by the scaling function $\varphi(x)$.

The scaling function itself is a wavelet (sometimes called a *father* wavelet), since $\{\varphi_{j,k}\}_{j,k\in\mathbb{Z}}$ forms an orthonormal basis for $L^2(\mathbb{R})$, where $\varphi_{j,k} := 2^{j/2}\varphi(2^j x - k)$. The Daubechies father wavelets are specified by its filter coefficients. For each integer $N \geq 1$ there is an orthogonal MRA that generates compactly supported wavelets, with length of support $2N$. These wavelets are the Daubechies father wavelets, which are specified by $2N$ filter coefficients. In this paper these wavelets are referred to as the Daubechies N wavelet. The Daubechies 1 wavelet is the Haar wavelet. The Daubechies 4 wavelet, used in section 4, has filter coefficients

$$\begin{array}{llll} h_0 \approx 0.2304, & h_1 \approx 0.7148, & h_2 \approx 0.6309, & h_3 \approx -0.0280, \\ h_4 \approx -0.1870, & h_5 \approx 0.0308, & h_6 \approx 0.0329, & h_7 \approx 0.0106. \end{array}$$

Given an orthogonal MRA with approximation subspaces, $\{V_j\}_{j\in\mathbb{Z}}$, the *orthogonal complement of* V_j *in* V_{j+1} is denoted W_j. That is, W_j is the subspace of $L^2(\mathbb{R})$ consisting of vectors in V_{j+1} that are orthogonal to vectors in V_j. The subspace W_j is called the *detail* subspace at scale 2^{-j}. By definition $V_j \perp W_j$ and thus for any $f \in V_{j+1}$, there exists unique $g \in V_j$ and $h \in W_j$, such that $f = g + h$. The function g is the orthogonal projection of f onto V_j. Similarly, h is the orthogonal projection of f onto W_j.

Since $\{\varphi_{j,k}\}_{k\in\mathbb{Z}}$ forms an orthonormal basis for V_j, we can write

$$g = P_j f := \sum_{k\in\mathbb{Z}} \langle f, \varphi_{j,k}\rangle \varphi_{j,k}(x).$$

The projection $P_j f$ is the best approximation of f in the subspace V_j. It can be show that the *difference operator*

$$Q_j f := P_{j+1} f - P_j f \tag{5}$$

is the orthogonal projection of f onto W_j, hence $h = Q_j f$. To recover $P_{j+1} f$ we add $P_j f$ to the difference operator $Q_j f$. In other words, for a better approximation to f, $P_{j+1} f \in V_{j+1}$, one adds the details $Q_j f \in W_j$ to the approximation $P_j f \in V_j$. So, for each $j \in \mathbb{Z}$,

$$V_{j+1} = V_j \oplus W_j.$$

The difference operator $Q_j f$ can also be written in terms of basis functions for W_j. The basis functions for W_j are actually $\{\psi_{j,k}\}_{k\in\mathbb{Z}}$, so

$$Q_j f = \sum_{k\in\mathbb{Z}} \langle f, \psi_{j,k}\rangle \psi_{j,k}(x).$$

It is Mallat's Theorem [10] that demonstrates that φ determines a mother wavelet ψ and the family $\{\psi_{j,k}\}_{k\in\mathbb{Z}}$ forms an orthonormal basis for W_j.

Theorem 1. *Given an orthogonal MRA with scaling function φ, there is a wavelet $\psi \in L^2(\mathbb{R})$ such that for each $j \in \mathbb{Z}$, the family $\{\psi_{j,k}\}_{k\in\mathbb{Z}}$ is an orthonormal basis for W_j. Hence the family $\{\psi_{j,k}\}_{j,k\in\mathbb{Z}}$ is an orthonormal basis for $L^2(\mathbb{R})$.*

Some wavelets do not come from an MRA. If the wavelet has compact support it does come from an MRA. Theorem 1 tells us there is a way to move from the father wavelet φ to the mother wavelet ψ. For example, the Daubechies 4 mother wavelet in figure 1b can be determined from the filter coefficients of the father wavelet, $h_0, h_1, \dots, h_7$.

2.2 Two-Dimensional Wavelets

Now that the multiresolution structure of the one-dimensional wavelet is understood, it is easier to understand the multiresolution structure of two-dimensional wavelets. The important idea from the one-dimensional case is that $V_{j+1} = V_j \oplus W_j$. In words, to obtain a finer approximation in V_{j+1}, details in W_j are added to the current approximation belonging to V_j. Decomposing an image into approximations and details is beneficial for image compression.

A standard approach to construct bases for a two-dimensional space from one-dimensional bases is the tensor product. Given a wavelet basis $\{\psi_{j,k}\}_{j,k\in\mathbb{Z}}$ in $L^2(\mathbb{R})$, the family of tensor products

$$\psi_{j,k;i,n}(x,y) = \psi_{j,k}(x)\psi_{i,n}(y), \quad \text{for } j,k,i,n \in \mathbb{Z},$$

is an orthonormal basis in $L^2(\mathbb{R}^2)$. This standard approach is not appropriate for two-dimensional wavelet decomposition since the multiresolution structure is lost; scales in $\psi_{j,k;i,n}(x,y)$ are mixed up since scaling parameters i, j need not be related.

The fix is simple since $\{\psi_{j,k}\}_{k\in\mathbb{Z}}$ is an orthogonal basis for V_j. We use the above idea at the level of the approximation spaces V_j. Let $\mathcal{V}_j$ be the closure in $L^2(\mathbb{R}^2)$ of the linear span of the tensor product $\psi_{j,k,n}(x,y) = \psi_{j,k}(x)\psi_{j,n}(y)$. That is,

$$\mathcal{V}_j = V_j \otimes V_j := \left\{ f(x,y) = \sum_{n,k} a_{j,k,n}\psi_{j,k,n}(x,y) : \sum_{n,k} |a_{j,k,n}|^2 < \infty \right\}.$$

Subspaces $\mathcal{V}_j$ form an MRA in $L^2(\mathbb{R}^2)$. The scaling function is $\psi(x,y) = \psi(x)\psi(y)$ and $\{\psi(x-k, y-k)\}_{k,n\in\mathbb{Z}}$ form an orthogonal basis for $\mathcal{V}_0$. Movement between subspaces $\mathcal{V}_j$ and $\mathcal{V}_{j+1}$ is done by scaling by two in both variables x and y. Note that the bases constructed for each $\mathcal{V}_j$ are separable. Therefore implementation can easily be done using the one-dimensional Fast Wavelet Transform in each dimension.

Denote the orthogonal complement of $\mathcal{V}_j$ in $\mathcal{V}_{j+1}$ with $\mathcal{W}_j$. Three wavelets are needed to span the detail space $\mathcal{W}_j$ since

$$\begin{aligned}\mathcal{V}_{j+1} &= (V_j \oplus W_j) \otimes (V_j \oplus W_j)\\ &= (V_j \otimes V_j) \oplus [(V_j \otimes W_j) \oplus (W_j \otimes V_j) \oplus (W_j \otimes W_j)]\\ &= \mathcal{V}_j \oplus \mathcal{W}_j.\end{aligned}$$

So $\mathcal{W}_j$ is the direct sum of three tensor products, requiring three wavelets to specify it. These three wavelets are

$$\psi^d(x,y) = \psi(x)\psi(y), \quad \psi^v(x,y) = \psi(x)\varphi(y), \quad \psi^h(x,y) = \varphi(x)\psi(y),$$

where d, v and h stand for diagonal, vertical and horizontal, respectively. The wavelet ψ^d favors details in the diagonal direction, ψ^v favors details in the vertical direction and ψ^h favors details in the horizontal direction.

These favorable directions can easily be seen for the two-dimensional Haar basis. The scaling function is given by $\varphi(x,y) = \chi_{[0,1)}(x)\chi_{[0,1)}(y)$, which can be depicted, along with the Haar mother wavelets, as

$$\begin{array}{cccc} \varphi(x,y) & \psi^d(x,y) & \psi^v(x,y) & \psi^h(x,y) \\ \begin{array}{|c|c|}\hline 1 & 1 \\ \hline 1 & 1 \\ \hline\end{array} & \begin{array}{|c|c|}\hline -1 & 1 \\ \hline 1 & -1 \\ \hline\end{array} & \begin{array}{|c|c|}\hline 1 & -1 \\ \hline 1 & -1 \\ \hline\end{array} & \begin{array}{|c|c|}\hline -1 & -1 \\ \hline 1 & 1 \\ \hline\end{array} \end{array}.$$

An example of the decomposition of the well-known Lena image is given in figure 2. We suppose the fine resolution Lena image lives in $\mathcal{V}_2$, which means the image is the approximation a_2. The first level of decomposition obtains the coarser approximation $a_1 \in \mathcal{V}_1$ using 1/4 of the data. Also during the first level of decomposition the details d_1^h, d_1^d, $d_1^v \in \mathcal{W}_1$ are computed. The same process is repeated for a_1 in the second level of decomposition yielding $a_0 \in \mathcal{V}_0$ and d_0^h, d_0^d, $d_0^v \in \mathcal{W}_0$. Note that the size of the original image does not change after each decomposition level. The approximation image a_j just becomes 1/4 the size of the previous a_{j-1}. This decomposition of the image into approximations and details is used next as part of the compression algorithms.

3 Wavelet Based Compression Algorithms

Here we discuss the overall steps of image compression and detail the baseline and EZW algorithms. We discuss only grayscale images where each pixel has grayscale value between 0 and 255. There are three main components to image compression: transform, quantize and encode. Applying a transform should remove some of the redundancy in the data by hopefully resulting in mostly zero values. Quantization reduces the precision of the transformed data,

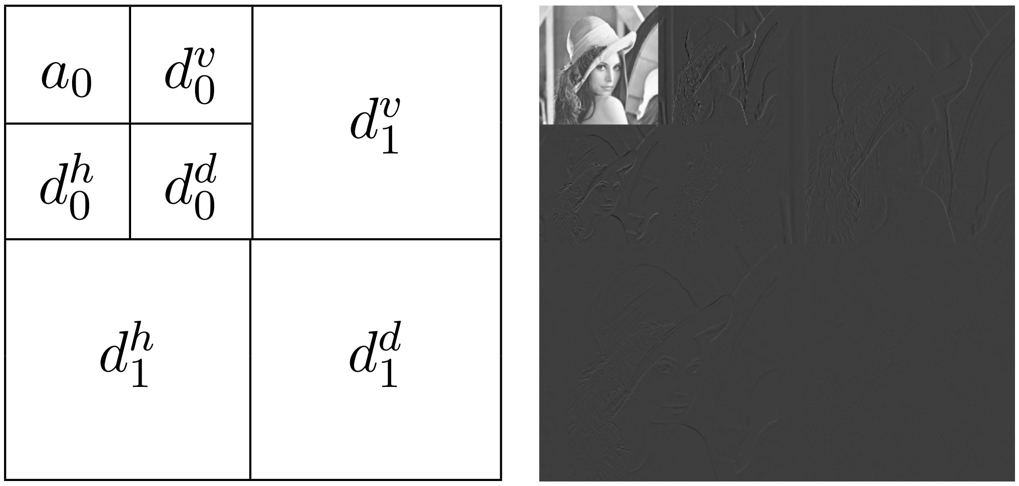

a_0
d_0^v
d_1^v
d_0^h
d_0^d
d_1^h
d_1^d
a) Schematic of decomposition
b) Actual decomposition of image

Figure 2: Two-dimensional wavelet decomposition of Lena image after 2 levels of decomposition. The detail spaces have coefficients near zero, but pick out fine details such as edges in the image.

which leads to rounding errors. The final step of encoding aims to store the quantized data compactly.

In this paper the transformation step is done using a wavelet transform. It has been suggested that the wavelet transform acts similar to human visual system [6]. A wavelet transform results in many zero or near zero grayscale values. A thresholding step is usually applied to the transformed data, that is any coefficient smaller than some prescribed value is set to zero. Thresholding is acceptable since many values near zero represent insignificant data to the human visual system. The level of thresholding, T, is a parameter that can be adjusted by the user.

Quantization is the only step that loses information. By reducing the precision of the wavelet transform rounding errors are incurred and information is lost. This step is of course necessary for effective compression of the data. One simple approach to quantization is to represent the wavelet transform with nearest integer values. Integers are stored using a smaller number of bits than floating point numbers. The amount of storage is therefore decreased.

Location of nonzero values in the quantized data is of importance in compression. Encoding the data tries to represent the location of nonzeros compactly. The relationship between the quantized data and encoding is the crucial aspect of wavelet based image compression. Each wavelet based compression algorithms explores this relationship differently.

3.1 Baseline Wavelet Compression Algorithm

The baseline algorithm is quite simple compared to the EZW algorithm. This description follows that of Weeks [5]. There are some problems with this baseline algorithm that other more advanced methods have aimed to correct. It is difficult to determine the exact compression rate or the exact error in compression, for example. The baseline algorithm also does not allow for progressive transmission. That is, with this algorithm one can not send successive data packets (over the Internet) that produce better and better resolution of the image. In spite of these defects, the simplicity of this algorithm provides an excellent baseline for comparison.

Like all wavelet based image compression algorithms, a two dimensional wavelet transform of the image is computed first. A threshold is then applied to the coefficients $Wf(j,k)$ to obtain a significance map

$$s(j,k) = \begin{cases} 0, & \text{if } |Wf(j,k)| < T, \\ 1, & \text{if } |Wf(j,k)| \geq T. \end{cases}$$

The map $s(j,k)$ provides a way to represent the location of the values of $Wf(j,k)$ that are visually significant. The $Wf(j,k)$ of significance are then rounded to their nearest integer value. By rounding the $Wf(j,k)$ a compression is obtained since integers can be stored with less bits than floating point numbers.

To achieve more compression the rounded $Wf(j,k)$ values are stored compactly. The encoding method used here is Huffman coding [5]. From input data, Huffman coding creates a list of distinct values and then determines the multiplicity of each. A binary tree is constructed with the distinct values as leaves. The values that have larger multiplicity are closer to the root of the tree. A left branch is represented as 0 and a right branch is denoted 1. Encoding the rounded coefficients of $Wf(j,k)$ is simply done by finding its value in the tree. For example, the code 101 encodes the value found by going down a right branch, then a left branch and one more right branch. Recognize that the more frequent a value occurs, the fewer number of bits needed to represent it. Numerical examples with this baseline algorithm are given in section 4, while the EZW method is discussed next.

3.2 Embedded Zerotree Wavelet Algorithm

The embedded zerotree wavelet (EZW) algorithm is more complicated then the baseline above. The description here follows that of Walker and Nguyen [6]. The complexity of the EZW algorithm is justified since it remedies the two deficiencies mentioned above for the baseline algorithm. That is, the EZW algorithm has the ability to specify a compression rate or a desired error. The EZW algorithm can also progressively transmit successive resolutions of the image.

The EZW algorithm was one of the first wavelet based compression algorithms to show the power of using wavelets. Shapiro [7] first described the EZW algorithm in 1993. The method introduces fundamental concepts that more superior methods utilize. One now scans through the wavelet coefficients in a strategic manner when determining $s(j,k)$. Namely, zigzag through approximation spaces and diagonal details, column scan through vertical details and row scan through horizontal details. Figure 3 depicts the scan order for 2-level and 3-level transforms of an 8 by 8 pixel image. This scan order concept could easily be applied to the baseline algorithm as well.

The next step is quantization. Quantization is accomplished using an embedded coding. An embedded coding allows for progressive transmission of the coefficients $Wf(j,k)$. Using

1	2	5	8	17	24	25	32
3	4	6	7	18	23	26	31
9	10	13	14	19	22	27	30
12	11	15	16	20	21	28	29
33	34	35	36	49	50	54	55
40	39	38	37	51	53	56	61
41	42	43	44	52	57	69	62
48	47	46	45	58	59	63	64

a) 2-level transform

1	2	5	8	17	24	25	32
3	4	6	7	18	23	26	31
9	10	13	14	19	22	27	30
12	11	15	16	20	21	28	29
33	34	35	36	49	50	54	55
40	39	38	37	51	53	56	61
41	42	43	44	52	57	69	62
48	47	46	45	58	59	63	64

b) 3-level transform

Figure 3: Representation of scan order for an 8 by 8 pixel image. The scan order for the 2-level and 3-level transforms are given in a) and b), respectively.

an embedded coding for quantization allows the user to recieve more accuracy in $Wf(j,k)$ without running the algorithm again from the start. The embedded coding used by the EZW algorithm is called bit-plane coding. Bit-plane encoding consists of the five steps given in Algorithm 1.

An example will help describe the steps of Algorithm 1. Suppose we have just two coefficients $w = -9.5$ and $v = 42$. Consider setting the initial threshold as $T_0 = 64$. During the first loop $T_1 = 32$, the output is the sign of v and quantized coefficient magnitudes $w_Q = 0$ and $v_Q = 32$. Second loop has $T_2 = 16$, the refinement pass produces the bit 0 since $v \in [32, 32 + 16)$. No output is obtained from the significance pass of the second loop. The quantized coefficient magnitudes are still $w_Q = 0$ and $v_Q = 32$. The third loop with $T_3 = 8$, the significance pass outputs the sign of w. The refinement pass outputs the bit 1 because $v \in [32 + 8, 32 + 16)$. The quantized coefficient magnitudes are $w_Q = 8$ and $w_Q = 40$.

The maximum error between the coefficient and the quantized coefficient is less than $T_0/2^i$, after i loops. The ordering of this encoding, highest magnitude bits being encoded first, allows for progressive transmission. The wavelet transform is partically well-suited for bit-plane encoding since the wavelet transform of natural scene images have relatively few high magnitude values. These high magnitude coefficients are coarsely approximated during the first few loops of bit-plane encoding. This produces a low resolution image. Subsequent loops can be computed to encode lower magnitude coefficients and refine the high magnitude coefficients. This will produce higher resolution images of the original. Bit-plane encoding loops are continued until a given bit budget is exhausted or a given error is achieved.

Now that the coefficients are quantized by the bit-plane encoding, *zerotrees* are used to compactly represent the insignificant coefficient values. A zerotree is a *quadtree*, which for a given threshold T, has insignificant wavelet coefficients at *all* of its locations. A quadtree is a tree of wavelet coefficient locations with *root* at $[i, j]$ and *children* located at $[2i, 2j]$, $[2i+1, 2j]$, $[2i, 2j + 1]$ and $[2i + 1, 2j + 1]$, and each of their children, and so on. The *descendents* of the root go all the way back to the first level of wavelet decomposition. Figure 4 shows two

Algorithm 1 — Bit-plane encoding

```
1. Initialize. Choose initial threshold, $T = T_0$, so that all coefficients $|Wf(j,k)| < T_0$ and
at least one satisfies $|Wf(j,k)| \geq T_0/2$.
2. Update threshold. Let $T_m = T_{m-1}/2$.
3. Significance pass. Scan through insignificant $Wf(j,k)$ using the scan order introduced
above. Test each value $Wf(j,k)$ as follows:

if $|Wf(j,k)| \geq T_m$ then
  - Output sign of $Wf(j,k)$.
  - Set $|W_Q f(j,k)| = T_m$.
else if $|Wf(j,k)| < T_m$ then
  - Let $W_Q f(j,k)$ retain intial value of 0.
end if

4. Refinement pass (for $m > 1$). Scan through significant $Wf(j,k)$ found with higher
threshold values $T_n$, for $n < m$. For each significant $Wf(j,k)$:

if $|Wf(j,k)| \in [W_Q f(j,k), W_Q f(j,k) + T_m)$ then
  - Output bit 0.
else if $|Wf(j,k)| \in [W_Q f(j,k), W_Q f(j,k) + T_m)$ then
  - Output bit 1.
  - Replace $W_Q f(j,k)$ by $W_Q f(j,k) + T_m$.
end if

5. Loop. Repeat steps 2-4.
```

1	2	5	8	17	24	25	32
3	**4**	6	7	18	23	26	31
9	10	**13**	**14**	19	22	27	30
12	11	**15**	**16**	20	21	28	29
33	34	35	36	**49**	**50**	**54**	**55**
40	39	38	37	**51**	**53**	**56**	**61**
41	**42**	43	44	**52**	**57**	**69**	**62**
48	**47**	46	45	**58**	**59**	**63**	**64**

Figure 4: Two quadtrees are shown in bold, one with root 4 and another with root 12.

quadtrees in bold, one with root 12 at $[4, 1]$ and children $\{41, 42, 47, 48\}$. Another quadtree has root 4 at $[2, 2]$ and children $\{12, 14, 15, 16\}$ and their children $\{49, 50, \dots 64\}$.

Zerotrees can give a very compact description of the locations of insignificant coefficients. It is only necessary to encode one symbol, say R, to denote the root of a zerotree. The decoder can then (after transmission) locate all other insignificant coefficients in the zerotree. Obviously these zerotrees are only useful if they occur frequently. With the wavelet transform of natural scenes, it is fortunate that the multiresolution structure produces many zerotrees [6]. The zerotrees are incorporated in the EZW compression algorithm through the significance pass (step 3 of Algorithm 1). The `if` statement in step 3 is coded using zerotrees, which produce smaller number of bits. For more information on zerotrees, a rigorous and statistical discussion can be found in [7].

A highly refined version of EZW is the *set partitioning in hierarchical trees* (SPIHT) algorithm. The SPIHT algorithm was first introduced by Said and Pearlman [8]. Some of the highest peak signal to noise ratios (PSNR) are obtained with SPIHT (a definition of PSNR is given in section 4). Due to the performance of SPIHT, it is one of the most widely used wavelet based compression algorithms. The full algorithm is described in [6], but is not discussed here. The SPIHT algorithm is more tedious than EZW, however, it builds from the same concepts. Numerical results with the SPIHT will also be given in section 4.

4 Numerical Results

Now that the methods have been explained, some numerical examples of a real image are given. These numerical experiments are carried out using the popular example in image compression, Lena. Implementation of the above methods (and SPIHT) are done in MATLAB using the `wcompress` function. The `wcompress` function has a parameter that is used to specify the wavelet compression method. The three parameter values `gbl_mmc_h`, `ezw` and `spiht` were specified for the baseline, EZW and SPIHT compression methods, respectively. Each method is used to compress the Lena image with two different wavelets, the Haar wavelet and Daubechies 4 wavelet.

The goal of compression is to obtain a large compression ratio (CR) while maintaining acceptable image quality. The CR is a measure between the size of the original image and the compressed image. MATLAB measures the CR in terms of a percentage of the original image size. Usually the CR is reported as the ratio of the original size to the compressed size. For example, CR = 2:1 means that the original image is twice the size of the compressed image. Since MATLAB uses a different definition for CR care has to be taken during implementation.

To examine the quality of compression each image is decompressed. The decompression is done for visualization of the image. However, when one transmits (or stores) the image it would be transmitted as the compressed image. Then the image is decompressed after transmission. The wavelet transform itself is invertible, however information is lossed during the quantization step. Therefore differences in the original and compressed/decompressed image are expected.

Results from the baseline algorithm are difficult to compare to results of EZW and SPIHT. Due to the progressive transmission ability (inability) of each method, different parameters are tuned for compression. The baseline algorithm is included in this paper to help understand the basic steps of image compression. Before the differences are discussed, the criteria that measures quality of compression are introduced.

These quality of compression measures are the mean squared error (MSE) and the peak signal to noise ratio (PSNR). The MSE is a common way to measure error, it gives the mean squared difference between each pixel of the image. That is, for two images f and g the MSE is defined as

$$\text{MSE} = \frac{1}{N}\sum_{j,k}(f[j,k]-g[j,k])^2, \tag{6}$$

where $[j,k]$ gives the location of one of the N pixels in each image. The more useful measure of quality is the PSNR. The PSNR is a logrithmic measure, which seems to correspond with how people perceive intensity. Between two grayscale images with 256 graylevels we have

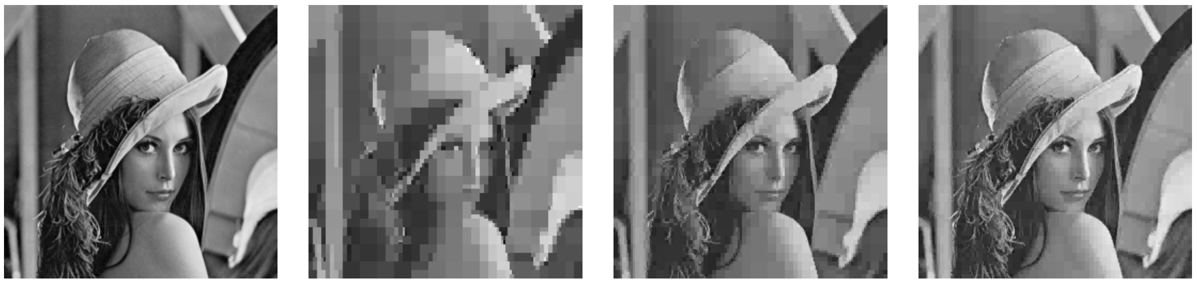
$$\text{PSNR} = 10\log_{10}\left(\frac{255^2}{\text{MSE}}\right), \tag{7}$$

which gives a measure in decibels (dB). In general, with a PSNR greater than 40 dB the compressed image is virtually the same to human observers.

The baseline algorithm differs from the EZW and SPIHT algorithms in that it cannot preform progressive transmission. The baseline algorithm uses thresholding, quantization and Huffman encoding as mentioned in section 3. It is difficult to determine the exact CR before compression is performed. Matlab does however approximate the threshold needed to obtain a desired CR. Therefore in the implementation of the baseline algorithm one specifies a desired CR as a parameter. The compression is then performed and a CR closed to the desired one is achieved. How close the desired CR is to the actual CR is difficult to quantify for general images.

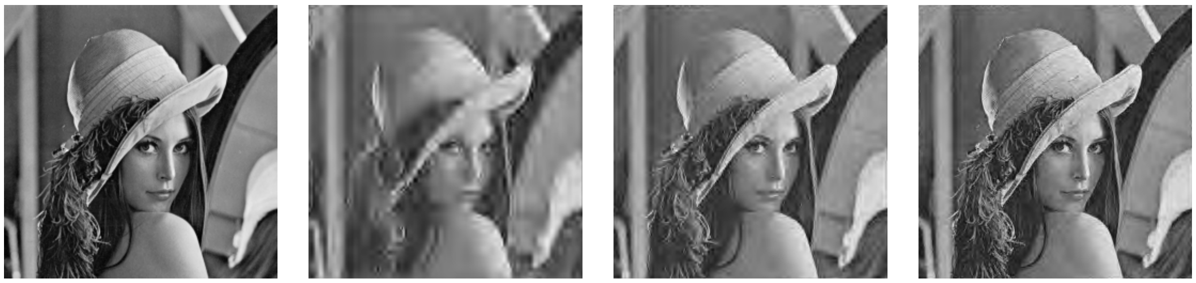
Compression results for the baseline algorithm on the Lena image are given in figures 5 and 6. Figure 5 is the baseline algorithm using the Haar wavelet, while figure 6 is with the Daubechies 4 wavelet. Each figure shows compression for different desired CR, namely 57:1, 20:1 and 10:1. These CR values were chosen to be comparable with the EZW algorithm. Notice that the actual CR are not exactly the specified desired values.

As expected the CR decreases as PSNR increases. It is found that if the CR value is shrunk more, then the PSNR does not become any better than ≈ 30 dB. This is believed to



a) Original b) CR = 50.0:1 PSNR = 22.2 dB c) CR = 16.3:1 PSNR = 27.3 dB d) CR = 8.4:1 PSNR = 30.7 dB

Figure 5: Baseline algorithm compression results for the Lena image using Haar wavelet.



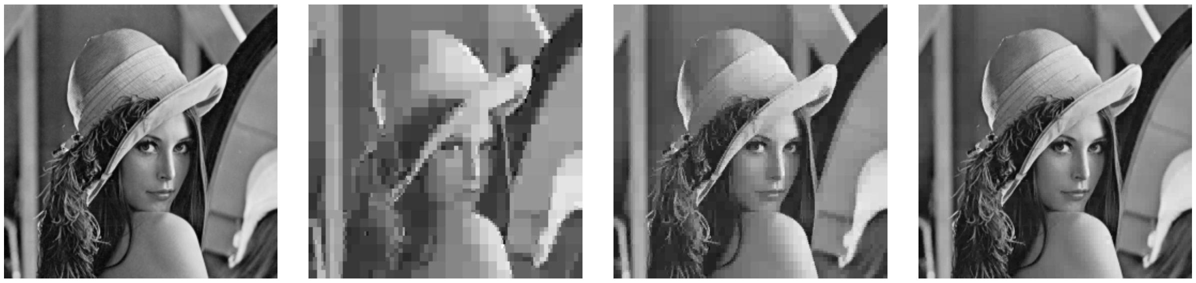
a) Original | b) CR = 31.2:1 PSNR = 23.4 dB | c) CR = 18.2:1 PSNR = 28.5 dB | d) CR = 9.6:1 PSNR = 30.6 dB

Figure 6: Baseline algorithm compression results for the Lena image using Daubechies 4 wavelet.

be because the quantization method loses a certain amount of information, independent of the thresholding. Both the Haar wavelet and the Daubechies 4 wavelet obtain comparable results. At higher CR, the piecewise constant structure of the Haar wavelet is apparent (figure 5b). The continuous structure of the Daubechies 4 wavelet returns more natural looking compression at high CR (figure 6b). This difference between the Haar and Daubechies 4 wavelet can be seen with all compression algorithms. The Haar and Daubechies 4 wavelets do however obtain nearly the same PSNR.

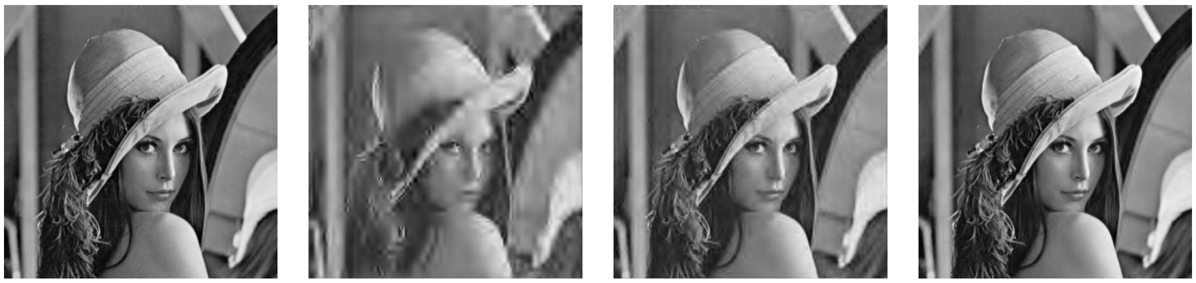
The EZW and SPIHT algorithms return compression results different from the baseline algorithm. In Matlab the EZW and SPIHT algorithms are implemented with stopping criterion as the number of loops. This shows the progressive transmission nature of the methods. The algorithms are used on a given image and produce a compressed image, which has a certain CR. If a higher quality compression is desired more loops of the algorithm are computed. This is why it is difficult to compare with the baseline algorithm. Since we used CR values close to that of EZW it is easier to compare with the baseline.

Figures 7 and 8 show compressions for the Lena image using the EZW algorithm. While compressions using the SPIHT algorithm are show in figures 9 and 10. The number of loops



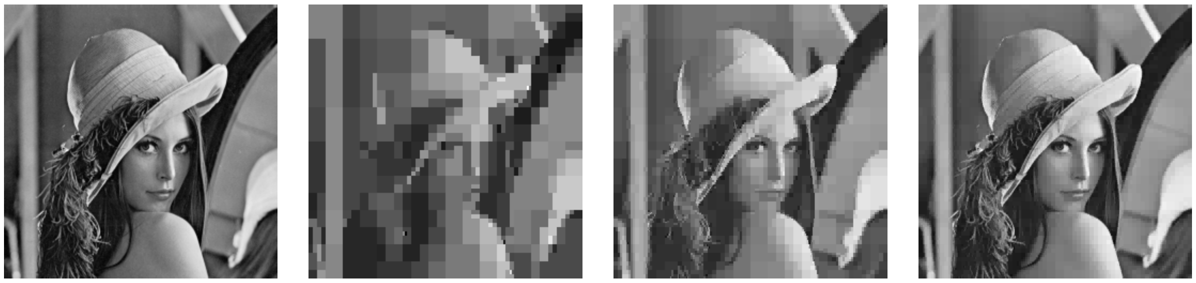
a) Original | b) CR = 51.6:1 PSNR = 22.8 dB | c) CR = 10.1:1 PSNR = 29.7 dB | d) CR = 3.1:1 PSNR = 38.3 dB

Figure 7: EZW compression results for the Lena image using Haar wavelet. The EZW algorithm is computed for 8, 10, and 12 loops in b), c) and d), respectively.



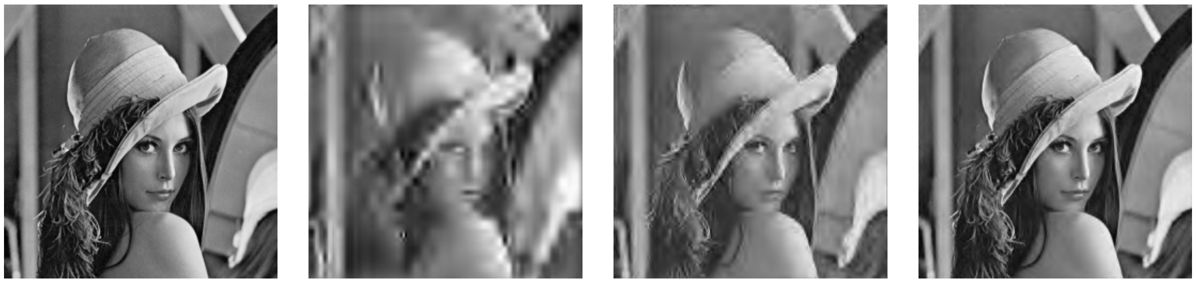
a) Original | b) CR = 58.6:1 PSNR = 23.5 dB | c) CR = 11.5:1 PSNR = 30.4 dB | d) CR = 3.5:1 PSNR = 38.6 dB

Figure 8: EZW compression results for the Lena image using Daubechies 4 wavelet. The EZW algorithm is computed for 8, 10, and 12 loops in b), c) and d), respectively.



a) Original | b) CR = 187.3:1 PSNR = 20.0 dB | c) CR = 31.6:1 PSNR = 25.4 dB | d) CR = 7.8:1 PSNR = 32.9 dB

Figure 9: SPIHT compression results for the Lena image using Haar wavelet. The SPIHT algorithm is computed for 8, 10, and 12 loops in b), c) and d), respectively.



a) Original | b) CR = 212.1:1 PSNR = 20.6 dB | c) CR = 38.3:1 PSNR = 26.0 dB | d) CR = 9.4:1 PSNR = 33.5 dB

Figure 10: SPIHT compression results for the Lena image using Daubechies 4 wavelet. The SPIHT algorithm is computed for 8, 10, and 12 loops in b), c) and d), respectively.

computed for both the EZW and SPIHT algorithm are 8, 10 and 12. The compressions have better PSNR and CR become smaller as the number of loops increase. The compressions still show the difference in wavelet structure between the Haar wavelet and Daubechies 4 wavelet at large CR. The Daubechies 4 wavelet compressions seem more desirable compared to Haar wavelets; compressed images look more natural when using the Daubechies 4 wavelet.

The EZW algorithm can obtain PSNR values near 40 dB, which makes the compression basically indistinguishable from the original. All PSNR values are better than the baseline algorithm's PSNR at nearly the same CR. The SPIHT algorithm out performs the EZW algorithm in some respect. The PSNR values obtained with the SPIHT algorithm are less than that of the EZW algorithm. In the same number of loops however, the SPIHT algorithm obtains much larger CR with these smaller, but comparable, PSNR values. With just a couple more loops of the algorithm, SPIHT obtains larger PSNR values than the EZW algorithm.

5 Discussion

Image compression is a successful application of wavelets. The multiresolution structure of wavelets mimic the human visual system. This produces compressions that are acceptable to human observers. All three algorithms discussed here have used the multiresolution structure of wavelets, some better than others.

Image compression algorithms apply three main steps: transformation, quantization and encoding. The wavelet transform was used in the baseline, EZW and SPIHT algorithms. The wavelet transform splits the image into approximations and details at different scales. The baseline algorithm only uses the multiresolution structure of wavelets in this transformation step. The EZW and SPIHT algorithms have more accurate quantization that can be progressively transmitted. These algorithms outperform the baseline algortihm, not only due to this superior quantization, but they exploit the multiresolution structure during the encoding step.

The numerical examples in section 4 compared the above three algorithms. The baseline algorithm is conceptually simple, but is found to only achieve PSNR no greater than ≈ 30 dB. The EZW algorithm can produce compressions with much better PSNR, around 40 dB. The compression ratios of the EZW algorithm are not as good as the SPIHT algorithm however. The SPIHT algorithm can achieve compression ratios more than double that of the EZW algorithm, with nearly the same PSNR.

There are more wavelet based compression algorithms than the three algorithms discussed in this paper. There are improvements that are possible for the baseline algorithm. One can use an adaptive thresholding technique. For example, take different threshold values, T, for the different wavelet transform regions a_j, d_j^v, d_j^d, d_j^h. Improvements of the EZW and SPIHT algorithms, for applications that need exact positions of significant values, are the WDR and ASWDR algorithms. WDR stands for Wavelet Difference Reduction and ASWDR stands for Adaptively Scanned Wavelet Difference Reduction. For more information on these other algorithms see [6].

References

[1] M. C. Pereyra and L. A. Ward, *Harmonic Analysis: From Fourier to Wavelets.* American Mathematical Soc., 2012, vol. 63.

[2] K. Amaratunga, J. R. Williams, S. Qian, and J. Weiss, "Wavelet–galerkin solutions for one-dimensional partial differential equations," *International Journal for Numerical Methods in Engineering*, vol. 37, no. 16, pp. 2703–2716, 1994.

[3] P. Flandrin, "Wavelet analysis and synthesis of fractional brownian motion," *Information Theory, IEEE Transactions on*, vol. 38, no. 2, pp. 910–917, 1992.

[4] C. R. Johnson, E. Hendriks, I. J. Berezhnoy, E. Brevdo, S. M. Hughes, I. Daubechies, J. Li, E. Postma, and J. Z. Wang, "Image processing for artist identification," *Signal Processing Magazine, IEEE*, vol. 25, no. 4, pp. 37–48, 2008.

[5] M. Weeks, *Digital Signal Processing Using MATLAB & Wavelets.* Jones & Bartlett Learning, 2010.

[6] J. S. Walker and T. Q. Nguyen, "Wavelet-based image compression," *Sub-chapter of CRC Press book: Transforms and Data Compression*, 2001.

[7] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *Signal Processing, IEEE Transactions on*, vol. 41, no. 12, pp. 3445–3462, 1993.

[8] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 6, no. 3, pp. 243–250, 1996.

[9] A. Haar, "Zur theorie der orthogonalen funktionensysteme," *Mathematische Annalen*, vol. 69, no. 3, pp. 331–371, 1910.

[10] S. G. Mallat, "Multiresolution approximations and wavelet orthonormal bases of $L^2(\mathbb{R})$," *Transactions of the American mathematical society*, vol. 315, no. 1, pp. 69–87, 1989.