

Orthogonal Wavelet Coefficient Precision and Fixed Point Representation

Michael Weeks and Qin Wang
Department of Computer Science, Georgia State University

Abstract: The Discrete Wavelet Transform (DWT) is an important transform with many signal processing applications, most notably, compression. Many architectures have been proposed to perform DWT, but few address the precision of the coefficients necessary to ensure perfect reconstruction. The goal of this work is to experimentally determine the precision of the filter coefficients (for an orthogonal wavelet) needed to compute the 2-D DWT without introducing round-off error via the filter.

Key words: wavelet transform, fixed-point precision

1. INTRODUCTION

Wavelet analysis applies to many different applications, such as digital communications, biomedical signal processing, medical imaging, matrix computation, digital signal compression, and video-conferencing [1][2]. Wavelets can be used to represent data in a way that reduces redundancy within the signal. Therefore, it can be compressed and stored in less space after the transform.

The Discrete Wavelet Transform (DWT) is a basis transformation which is discrete in time and scale. Though the DWT produces real (floating-point) values, it uses integer time and scale values. Often, a design uses fixed-point format for data values, since fixed-point systems are easy to implement. Fixed-point data values can be thought of as rational numbers, and by using rational numbers for filter coefficients, we are technically approximating the DWT. However, one requirement in this work is that perfect reconstruction of 2-D images is achieved experimentally. The focus is to determine the amount of precision needed if fixed-point arithmetic is used to compute the transform.

The next section covers the DWT background. The experiment follows in section 3, and the final section presents results and conclusions.

2. DWT BACKGROUND

The goal of any transform is to turn the information of a signal into coefficients that can be manipulated, stored, transmitted and finally used to reconstruct the signal. In the case of lossy compression, a reconstructed signal that is a close approximation to the original is desired. By squeezing and stretching wavelets, the wavelet transform can adapt to catch the high frequency and low frequency components through a process called multiresolution. Let $w(t)$ represent orthogonal wavelets. Instead of shifting along time and stretching in a continuous scale (as in the continuous wavelet transform), $w(t)$ moves discretely in time and shrinks by half for each scale. Hence, $w_{jk}(t) = w(2^j t - k)$, j and k representing scale and time respectively, and the signal function $f(t)$ can be decomposed into [1],

$$f(t) = \sum b_{jk} w_{jk}(t),$$

where $b_{jk} = \int f(t) w_{jk}(t) dt$

This is the Discrete Wavelet Transformation (DWT).

The relationship between filter coefficients and wavelets function is [7]:

$$\phi(t) = 2 \sum h(k) \phi(2t - k),$$

where $\phi(t)$ is called the scaling function

$$w(t) = 2 \sum d(k) \phi(2t - k),$$

where $w(t)$ is the wavelet function

Therefore, if we have filter coefficients, the scaling and corresponding wavelet functions can be determined, and vice-versa.

Figure 1 shows a 2-Dimensional DWT structure for 1 octave. The first filter bank transforms the signal horizontally into an average signal (from a low-pass filter, H0) and a detail signal (from a high-pass filter, H1). A pair of filters operate on the data horizontally, then vertically. This specifies 1 octave, also called a level of resolution. Downsampling, or discarding a filter's even-numbered output values, occurs after each filtering operation. After transmission, an inverse up-sampling operation inserts zeros back in the even-numbered positions, and the signals go into reconstructing filter banks, with another low (F0) and high pass filter (F1). Finally, the two channel signals are combined together for the synthesis.

The DWT can be computed quickly by the fast pyramid algorithm [5]. The algorithm gets its efficiency through downsampling, giving it a complexity of $O(N)$ [6]. Since a DWT filter only keeps half of the filter outputs, then only half need to be computed [8]. The scaling filters generate $N/2^j$ values (j is the octave number), but these are only used internally as they are inputs to the next pair of filters. The exception is the last octave. The maximum number of octaves is based on the input length, $J = \log_2(N)$, however, practical applications limit the number of octaves, typically to $J=3$.

The wavelet transform stores the details (wavelet coefficients), and puts the average signal (scaling coefficients) back through a filter bank, to generate the next octave's coefficients. This process, called multi-resolution, can be repeated up to $\log(N)$ times.

2.1 Fixed Point Versus Floating Point

When designing a wavelet transform processor, one question is how to treat the data and filter coefficient values, as fixed-point values, or floating point? Fixed point has the advantages of being easier to implement, requires less silicon area, and makes multiplications faster to perform. Floating point allows a greater range of numbers, though floating point numbers require 32 or 64 bits. Fixed-point numbers can be 8 to 32 bits (or more), possibly saving space in the multiplier.

The wavelet transform has compact support, and the coefficients have small sizes that do not take advantage of the floating-point number's wide range [9]. Also, using fixed-point numbers allows for a simpler design. In other words, the extra hardware that floating point requires is wasted on the DWT. A standard design for integer multiplications is the high speed Booth multiplier, as used by [9], [10]. DWT architectures of [10-12], are implemented with fixed-point computations. In the horizontal and vertical filters, decimal numbers including some bits for the fractional part represent the values. When the results leave the filters, the values are truncated to fixed format.

For multiresolution, the DWT coefficient's range will grow [10]. The increase is upper-bounded by approximately 2. An extra bit of precision for each transform octave or dimension will be needed. For a 1-D architecture, [10] also notes that 12 bits of precision are enough. In [13], it is noted that wavelet transformed data is naturally floating point, but they round it to the nearest integer values, to minimize data loss. The chip in [14] has 16x16 size multipliers, but keep 16 bits of the result, which is chosen by software. Therefore, fixed-point

numbers with an assumed radix have been shown to work for computing the DWT.

2.2 Precision

When a computer represents wavelet filter coefficients, or any irrational number, it naturally imposes a certain precision on these quantities, introducing round-off error. With programs such as Matlab, it is easy to verify that one can achieve perfect reconstruction. Therefore, the question we are attempting to answer is not whether or not the filter values can be represented with less precision than actual floating point values, but what the cut-off is for perfect reconstruction when using rational numbers.

Nine grey-scale images were used in this study, including Figure 2. These input values are 8 bits wide. If the filter coefficients are also 8 bits wide, then the arithmetic result for 1 filtering operation has 16 bits (from an 8 bit x 8 bit multiplication). If all multiplication bits are kept every time, then the size of the coefficients will grow by a factor of 3 just for 1 octave of the 2-D DWT. Therefore, some bits are truncated. Most previous work that address this use either 8 or 16 bits in their 1-D designs, e.g. [14]. Analysis on the word size for the 1-D DWT is shown in [10], and concludes that 32 bits is too much for the 1-D case. This paper explores precision needed for the 2-D DWT, experimentally, for the filter coefficients themselves.

2.3 Previous Results

The precision needed to store the transformed coefficients was found experimentally in [17] to be 13, 14, and 15 bits for 1, 2, and 3 octaves, respectively, for the Daubechies 4 (or 8) coefficient wavelet. The format includes 1 sign bit, 2 bits after the radix point, and the rest are used for the integer part.

The coefficients of this wavelet are orthogonal, meaning that the same coefficients (with reversed order and sign) are used for both the forward and inverse transform.

Similar ideas have been explored, such as using the Integer Wavelet Transform (IWT) for lossy/lossless compression [19]. In the IWT, filter outputs are rounded to integer values. The paper in [19] presented a model for quality loss. As [17] showed, at least 2 bits past the radix point are required. The approach we took in [17] was to represent the outputs as rational numbers. Since the coefficients are represented in binary, the denominator is always a power of 2. That is,

multiplying by 2^k , followed by truncation, followed by division by 2^k (or an equivalent k -bit shift to the right) has a similar effect as the IWT.

3. PRECISION EXPERIMENT

To experimentally find how much precision is necessary, 2-D images were transformed, with integer multiplications followed by bit-shifting. The transform uses 2-D convolution to achieve the functionality of DWT in Matlab (and double-checked against Matlab's built in 2-D DWT function). Within the procedure, a bit-shift scheme has been adopted to give fixed-point operation on the image data. Three different data processing methods are applied on the image data. The first one uses fixed-point operation on both DWT and IDWT, and the second applies fixed-point DWT on the input image but uses floating-point IDWT in the reconstruction. The last method considers the fact that during the fixed-point operation the data are usually truncated due to limited precision, hence in order to obtain more chance to have a perfect reconstruction, it uses the ceiling function rather than rounding in the integer value conversion.

The Daubechies wavelet with 4 coefficients are used in this study. First, the image was decomposed into 1 octave of resolution by fixed-point DWT, then we tried 3 different methods of reconstruction to get back the original image. The reconstructed image is compared with the original in terms of pixel magnitude to find out whether perfect reconstruction is achieved. Three 3 variant bit precisions: 12, 13 and 14-bits, are experimented on the set of test images. The tables below give the resulting error between the reconstructed image and the original one, depending upon which reconstruction method is used. The error value in the table is computed by finding the absolute value of the difference for each pixel, and summing them up over the whole image. From the error values for each method, we can tell how many bits are needed for the perfect reconstruction.

4. RESULTS AND CONCLUSION

Table 1 indicates how the precision affects the transform, namely that for perfect reconstruction, we need at least 14 bits for 1 octave of decomposition if both DWT and IDWT are fixed-point operation, while

13 bits are enough to come up with the same result if only the transform DWT is fixed-point. In comparison of the reconstructed image and the original one, an interesting thing shows up. No matter what method or bit precision is adopted, if there is any error in existence, the error for each pixel is only 1. This is because the cause of the error is truncation in the fixed-point operation, so it is anticipated that in the integer value conversion on the reconstructed image, the ceiling function should produce a better restored image than the round off function does. The result from applying the ceiling function in the last two columns in the table firmly demonstrates this. The error induced by using 12-bits of precision with the ceiling operator is much smaller than using the other two methods. Only 13 bits are necessary for a perfect reconstruction when adopting fixed-point operation for both DWT and IDWT, which is a 1 bit improvement compared to the results of the round-off operation.

The figures of 'dog on porch' image of Figure 2 show how the different methods can affect the reconstruction in the fixed-point operation. The upper-left is the original image, the upper-right is the pixel-error from 12-bit dual fixed-point operation, the lower-left is for 12-bit fixed-point DWT/floating-point IDWT, and the error from 12-bit dual fixed-operation with ceiling function is shown in the lower-right. The difference is represented by a black spot at each pixel, the result shows whatever method is in use the magnitude of the difference is only 1. Comparing the difference image to the original, we notice that the area where the error stands out is where the brighter spot happens to be. This is because higher magnitude pixel values are distorted more than lower magnitude ones, i.e. the dark pixels.

This paper shows that the amount of precision of the filter coefficients (that is, the number of bits to use) is an important parameter behind the design of a system involving the wavelet transform. For a 2-D DWT, the number of bits to use depends upon the method of data processing in reconstruction. For the test images used in this study, 14 bits of precision allows the DWT to be computed without affecting the reconstruction, while 13 bits can be sufficient for using float-point IDWT reconstruction or by ceiling in integer conversion. Since the integer values on pixels for an image are usually distributed among the range of 0 to 255, if these values are all able to be restored perfectly after the reconstruction with certain precision, it is confident to say the same precision will work for any grey-scale images.

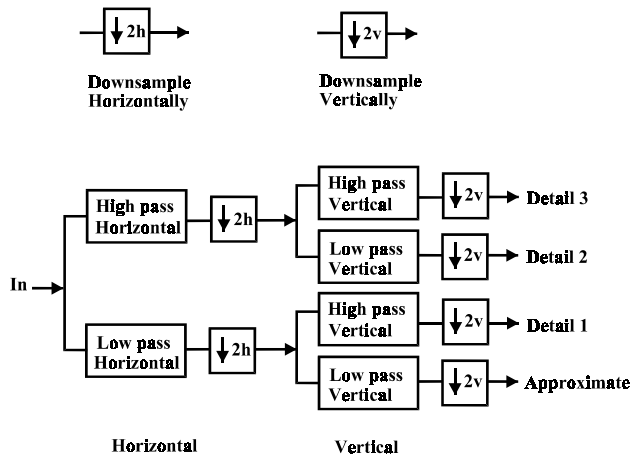


Figure 1. The 2-Dimensional DWT

Image name	size	fixed-point DWT/IDWT			floating-point IDWT		Ceiling to Integer	
		12 bit	13 bit	14 bit	12 bit	13/14 bit	12 bit	13/14 bit
airplane	256x256	16119	7487	0	15594	0	11188	0
dog	256x256	8103	532	0	5322	0	917	0
jelly_beans	256x256	14572	0	0	13290	0	7344	0
moon_surface	256x256	14079	54	0	8875	0	132	0
elaine	512x512	49330	4995	0	36837	0	10052	0
house	512x512	57953	7099	0	51267	0	25274	0
lena	512x512	45985	2766	0	34905	0	6948	0
tank	512x512	56450	0	0	46524	0	3	0

Table 1 – error values between the reconstructed and the original image for 3 different methods of reconstruction with 3 variant bit precisions

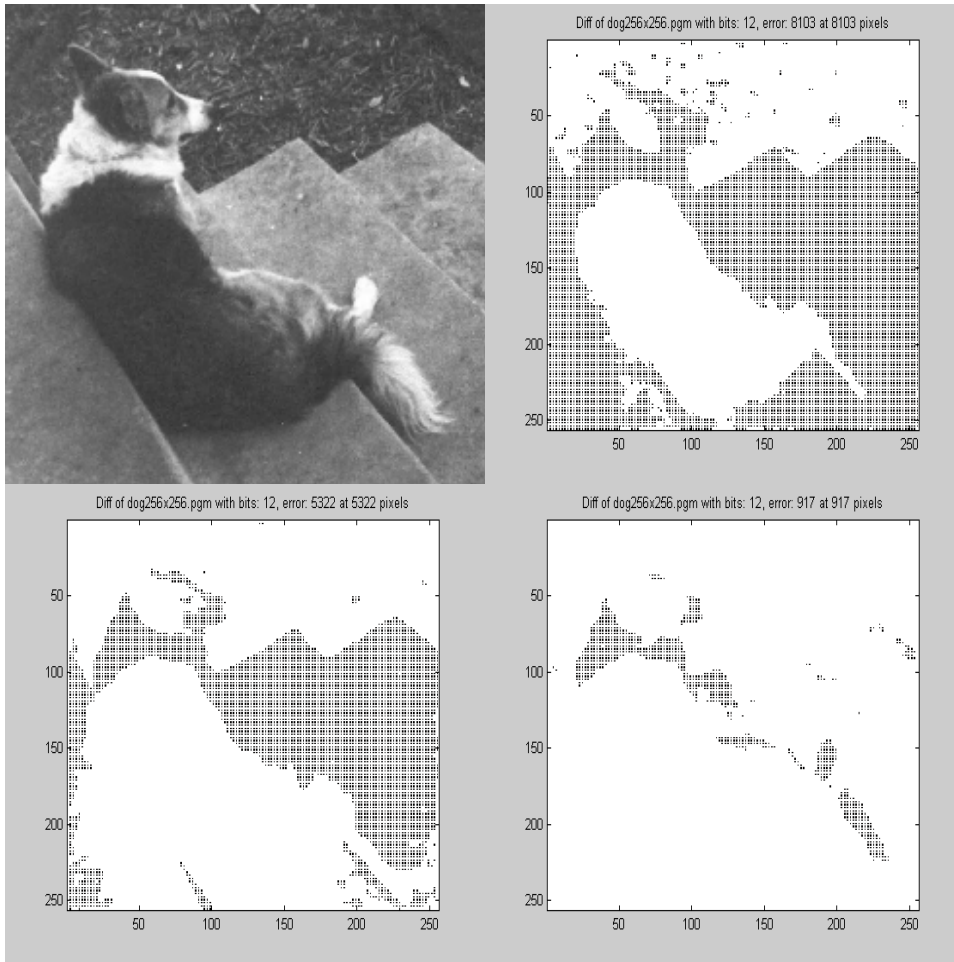


Figure 2 – Original image ‘dog256x256.pgm’ and 3 difference images from its corresponding reconstruction method. The up-left is original image, up-right is the pixel-error from 12-bit dual fixed-point operation, down-left is for 12-bit fixed-point DWT/float-point IDWT, and the error from 12-bit dual fixed-operation with ceiling function shown in the down-right.

REFERENCES

- [1] Andrew Bruce, David Donoho, and Hong-Ye Gao, "Wavelet Analysis," *IEEE Spectrum*, Oct. 1996, pages 26-35.
- [2] Mohan Vishwanath and Chaitali Chakrabarti, "A VLSI Architecture for Real-Time Hierarchical Encoding/Decoding of Video using the Wavelet Transform," *IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP '94)*, Adelaide, Australia, Volume 2, April 19-22, 1994, pages 401-404.
- [3] Keshab K. Parhi and Takao Nishitani, "VLSI Architectures for Discrete Wavelet Transforms," *IEEE Trans. VLSI Systems*, Vol. 1, No. 2, 1993, pages 191-202.
- [4] Tinku Acharya, Po-Yueh Chen, and Hamid Jafarkhani, "A Pipelined Architecture for Adaptive Image Compression using DWT and Spectral Classification," *Proc. of the Thirty-First Annual Conf. on Info. Sciences and Systems*, Volume II, Baltimore, MD, March 19-21, 1997, pages 1-17.
- [5] Stephane Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Patt. Anal. & Mach. Intel.*, Vol. 11, No. 7, 1989, pages 674-693.
- [6] M. Vishwanath, R. M. Owens, and M. J. Irwin, "Discrete Wavelet Transforms in VLSI," *Proc. of the Int. Conf. on Application Specific Array Processors*, Berkeley, CA, August 1-2, 1992, pages 218-229.
- [7] Gilbert Strang and Truong Nguyen, *Wavelets and Filter Banks*, Wellesley, MA, Wellesley-Cambridge Press, 1996.
- [8] G. Knowles, "VLSI Architecture for the Discrete Wavelet Transform," *Electronics Letters*, Vol. 26, No. 15, 1990, pages 1184-1185.
- [9] Charles C. Chang, Jyh-Charn Liu, and Andrew K. Chan, "On the Architectural Support for Fast Wavelet Transform," *SPIE Wavelet Applications IV*, Vol. 3078, Orlando, Florida, April 21-25, 1997, pages 700-707.
- [10] Aleksander Grzeszczak, Mrinal K. Mandal, Sethuraman Panchanathan, and Tet Yeap, "VLSI Implementation of Discrete Wavelet Transform," *IEEE Trans. VLSI Systems*, Vol. 4, No. 4, 1996, pages 421-433.
- [11] Jimmy Limquenco and Magdy Bayoumi, "A 2-D DWT Architecture," *Proc. 39th Midwest Symposium Circuits & Systems*, Ames, Iowa, August 18-21, 1996, pages 1239-1242.

[12] A. S. Lewis and G. Knowles, "VLSI Architecture for 2-D Daubechies Wavelet Transform without Multipliers," *Electronics Letters*, Vol. 27, No. 2, 1991, pages 171-173.

[13] Jun Wang and H. K. Huang, "Three-dimensional Medical Image Compression using a Wavelet Transform with Parallel Computing," *SPIE Imaging Physics*, San Diego, Vol. 2431, March 26-April 2, 1995, pages 16-26.

[14] *Wavelet Transform Processor Chip User's Guide*, Aware, Inc., Bedford, MA, 1994.

[15] Michael Weeks, Jimmy Limqueco, and Magdy Bayoumi, "On Block Architectures for Discrete Wavelet Transform," *32nd Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, November 1-4, 1998.

[16] Michael Weeks and Magdy Bayoumi, "3-D Discrete Wavelet Transform Architectures," *IEEE Int. Symp. Circuits & Systems (ISCAS '98)*, Monterey, CA, May 31 - June 3, 1998.

[17] Michael Weeks, "Precision for 2-D Discrete Wavelet Transform Processors," 2000 IEEE Workshop on Signal Processing Systems (SiPS), Lafayette, Louisiana, October 11-13, 2000, pages 80-89.