DATA COMPRESSION OF REMOTE SENSING IMAGES USING WAVELET TRANSFORMS

1Swetha V, 2John J P, 3Seetha Lakssshmi T.R
Asst. Professors, Department of CSE, Nagarjuna College of Engineering and Technology
Email: 1vuraswetha@gmail.com, 2johnypaul10@gmail.com, 3rseetha_1989@rediffmail.com

Abstract - Data Compression is efficient coding designed to correct overrepresentation that occurs in digital data handling systems. It finds innovative ways to represent information using as few bits as possible for storage and transmission. There are two types of compression techniques – lossless compression and lossy compression. In satellites, due to increase in the amount of data, it is proposed to implement data compression. Joint Photographic Experts Group (JPEG 2000), a Wavelet Transform based compression technique is used which mainly comprises of Wavelet Transformation, Quantization and Coding. This paper will provide implementation of Run-Length and Huffman Coding in JPEG 2000 for better compression ratios with improved image quality.

Index Terms – Wavelet Transform, Huffman Coding, Run-Length Coding

I. INTRODUCTION

As satellite technology becomes more sophisticated, the amount of data generated by the satellites increases exponentially. This increases bandwidth and storage space. So, we find a means of compressing the data without significant loss of information.

The generation of satellites implementing on-board data compression using the JPEG technique consists of three major components, the Discrete Cosine Transform (DCT), Quantization and Huffman Coding. The recent development in the field of wavelets has given rise to a wavelet-based data compression technique, known as JPEG 2000 like compression that offers superior performance as compared to the JPEG standard. It makes use of 2-D Discrete Wavelet Transform (DWT) as opposed to the use of DCT by JPEG standard; Quantization, Huffman and Run-Length Coding. We achieve development of wavelet-based data compression.

The images generated by the satellite cameras, as they scan the earth, are stored in the on-board memory storage devices in uncompressed format. The data is then transferred to the Peripheral Connect Interface (PCI) card for compression. Blocks are transferred to the Input Buffer on the PCI card. As it is not convenient to compress an entire block in one step, the block is split into smaller sized blocks, typically, 8X8 in size. Each block is then wavelet-transformed and quantized. Once the quantization is done, the transformed coefficients are encoded using Run-Length and Huffman Coding techniques. The above operation is done using the FPGA. Once the coding is complete, the data set is stored in the Output Buffer, from where it is read out as and when required. The entire operation is synchronized by the control logic circuitry.

Fig.1.Block Diagram of FPGA based Huffman and Run-Length Coding Computation Stage

A] Discrete Wavelet Transformation

The 8X8 blocks are Discrete Wavelet Transformed in this block. The Haar Transform is used for this operation. The result of this operation is an 8X8 block of transform coefficients. It is important to note that, till this point, no actual compression of data has been achieved. This transform operation merely serves to convert the original pixel values into transform coefficients. Most of the coefficients have small magnitudes and only a few coefficients have significant magnitude.

B] Quantizer

The output of DWT block is fed to the Quantizer. The Quantizer divides the transformed values by a number called the Quantization coefficient. The Quantization can be uniform or non-uniform in nature. The process of Quantization discards the coefficients that are small in magnitude and retains only the significant coefficients. This operation results in irretrievable loss of information.
However, it does not significantly impact the overall compression ratio.

C) Run-Length and Huffman Coding

Once the Quantization is done, the transform coefficients are encoded using the Run-Length and Huffman Coding technique. The Run-Length Coding and Huffman Coding techniques are used to further improve the compression ratio. The scope of this paper is limited to implementing the Run-Length and Huffman Coding (shown in hatched lines in Fig. 1.) for an 8 X 8 sized block in VHDL. The target platform in this work is from the Cyclone EP1C12 family, manufactured by the Altera Corporation. The tool used for the synthesis of the VHDL code is the Quartus II software from Altera Corporation.

II. WAVELETS

A wave is an oscillating function of time or space and is periodic. In contrast, wavelets are localized waves. A typical wave and wavelet are shown in Fig. 2. They have their energy concentrated in time or space and are suited to analysis of transient signals.

![Wave and Wavelet](image)

Fig. 2. Wave and Wavelet

A wavelet is a time domain signal that has compact support, i.e., limited in duration, and has an average value zero. It is usually oscillatory in nature and has most of its energy concentrated in a small amount of time. The word wavelet is due to Morlet and Grossman in the early 1980s. They used the French word ondelette - meaning "small wave". A little later it was transformed into English by translating "onde" into "wave" - giving wavelet.

A) Wavelet Analysis

Wavelet analysis aims to represent a signal using translations and dilations of a mother wavelet. Translation means shift in time and dilation means a shift in scale. To analyze a signal, a suitable wavelet is chosen and the correlation between the signal and the wavelet is calculated. The correlation coefficient depends on the similarity between the signal and the wavelet. Low-frequency and high-frequency information about the signal can be obtained by suitably scaling the wavelet. Wavelet analysis breaks up the image into several parts and then analyzes each part separately.

Wavelet analysis can be applied to n-dimensional signals. It is done by computing the Wavelet Transform of the signal under test. The transform of a signal is just another form of representing the signal. It does not change the information content present in the signal. The Wavelet Transform provides a time-frequency representation of the signal.

B) The Haar Transform

This paper uses Haar Wavelet which is the simplest type of wavelet. Haar transform has two important properties

- It conserves the energy of the image. This implies that the energy in the average values and difference values taken together is equal to the energy in the original image. Or, in other words, the energy in the transformed pixel values is equal to the energy in the original image.

- It performs compaction of energy. That is most of the energy in the image is concentrated in a few transformed coefficients.

C) Wavelet Method of Image Decomposition

The wavelet based method of image compression involves decomposition of an image into several subbands and then wavelet transforming each subband separately. There are seven possible ways of decomposition; each method has a different algorithm and gives different levels of energy compaction. Pyramidal decomposition is by far the most common method used to decompose the images.

III. CODING TECHNIQUES

Coding is a well-known algorithm designed with serial data in mind which is widely used in the compression of computer files to save disk space. An entropy encoding is a coding scheme that assigns codes to symbols so as to match code lengths with the probabilities of the symbols. Typically, entropy encoders are used to compress data by replacing symbols represented by equal-length codes with symbols represented by codes proportional to the negative logarithm of the probability. Therefore, the most common symbols use the shortest codes. Huffman coding is an entropy encoding algorithm used for lossless data compression that finds the optimal system of encoding strings based on the relative frequency of each character. A minimal variable-length character code is assigned to each character based on its frequency of occurrence in the data. It was developed by David A. Huffman as a Ph.D. student at MIT in 1952, and published in A Method for the Construction of Minimum-Redundancy Codes.

A) Huffman Coding

Huffman coding uses a specific method for choosing the representation for each symbol, resulting in a prefix-free code (that is, the bit string representing a particular symbol is never a prefix of a bit string representing any other symbol) where the frequently used characters are assigned smaller codes and less frequently used characters are assigned larger codes. It has been proven that Huffman coding is the most effective compression method of this type. No other mapping of source symbols to strings of bits will produce a smaller output when the actual symbol frequencies agree with those used to create the code.
Advantages of Huffman coding are that it is easy to understand, simple, has fairly quick encoding and decoding, high speed, good for variable length codes and lack of encumbrance by patents. In spite of being the most popular coding technique, Huffman coding has some disadvantages as computing Huffman tree is prohibitively expensive, allows codes to be represented in only integral number of bits, and so is somewhat less efficient. The above disadvantages can be overcome in arithmetic coding, where the codes can be represented in floating point numbers.

B) Run Length Coding

Run-Length Encoding (RLE) is a very simple form of data compression in which runs of data (that is, sequences in which the same data value occurs in many consecutive data elements) are stored as a single data value and count. Run-Length Encoding is supported by most bitmap file formats such as TIFF, BMP and PCX.

For example, consider a screen containing plain black text on a solid white background. There will be many long runs of white pixels in the blank space, and many short runs of black pixels within the text.

```plaintext
WWWBBWWWBBWWWBBWWWBBWWWBBWWWBBWWWBBWWWBBWWWBB
```

If we apply a simple run-length code to the above hypothetical scan line, we get the following:

12 W 12 B 14 W 24 W

Advantages and Disadvantages

RLE can compress any type of data regardless of its information content, but the content of data to be compressed affects the compression ratio. RLE compression is only efficient with files that contain lots of repetitive data. It cannot achieve high compression ratios compared to other compression methods, but it is easy to implement and is quick to execute.

IV. METHODOLOGY OF IMPLEMENTATION

Remote sensing images have varied pixel sizes. As it is not convenient to compress an entire image in one step, it is split into smaller sized blocks, typically, 8X8 in size. This also helps in speeding up calculations. The 8X8 block is first wavelet transformed, quantized and then rounded off. The resultant matrix is to be Run-Length and Huffman coded to get the compressed image. Each of the non-zero transform coefficients is given a Run/Cat event which in turn determines the Huffman code. Run is the number of transform coefficients equal to zero between the coefficient to be coded and the previous one not equal to zero in the same sequence and Category gives the number where the non-zero transform coefficient lies.

The Huffman code concatenated with the sign (if the coefficient is positive, it is represented as ‘0’ otherwise as ‘1’) and offset (distance of the transform coefficient from the mid value) gives the resultant compressed bits. These bits are grouped into byte format and the compressed code can be viewed in hexadecimal format. This constitutes the manual calculation and the compression ratio can be determined as Original number of bits / Number of bits after compression.

Coming to the design implementation, the inputs to the design are clk, opclk and i. ‘clk’ is used to synchronize the entire program. ‘opclk’ (8 times the period of the universal clock) is used to generate the output code in byte format. ‘i’ contains the 64 transform coefficients of an 8X8 sub-block of an input image which is to be compressed. Each of the non-zero transform coefficients is given a run/cat event which in turn determines the Huffman code. The Huffman code concatenated with the sign and offset gives the ‘output data’. ‘active’ signal determines the number of clock pulses (‘outclk’) to be generated for each non-zero transform coefficient. ‘output data’, with the assertion of the ‘outclk’ is sent to a parallel-in-serial-out register and then through a serial-in-parallel-out register to determine the output compressed code (qout) in byte format. Lastly, the compression ratio is computed as mentioned in manual calculation.

V. RESULTS

Two 512X512 pixel monochromatic images of an area in Bangladesh and Andhra Pradesh taken by the MSS of Landsat satellite are shown in Fig.3 and Fig.4. Each image is divided into 8X8 blocks and then subjected to JPEG 2000 like compression design implementation.

![Image 1](image1.png)

Fig.3. Image of an area in Bangladesh
The above matrix contains 64 transform coefficients 'i' which are compressed using design implementation. The compressed code in byte format can be viewed as follows:

```
E3  9E  B6  B1
00110000 00010011 00101100 10110000
30  09  16  B0
10011001 01001000 10110000 00100111
99  24  B0  13
00010110 00001111 01111000 10011000
0A  07  78  98
00100000 10011001 00100000 10000000
20  99  20  80
```

The last code contains only 1 bit and to represent it into byte format, 7 zero bits are padded. The resultant compression ratio can be determined as:

\[
\text{Compression ratio} = \frac{\text{Original number of bits}}{\text{Number of bits after compression}} = \frac{512}{132} = 3.35
\]

Similarly, the pixel values of an 8X8 block of Fig.4 are shown in Table 3.

```
TABLE 3
DATA SET 2
48  42  55  51  42  51  51
48  55  51  51  42  48  51
39  39  39  39  36  36  36
51  55  55  51  48  42  44
45  45  45  45  45  45  45
39  45  45  45  38  58  58
42  45  42  51  58  48  48
39  39  48  48  48  51  51
```

The pixel values of an 8X8 block of Fig.3 are shown in Table 1.

```
TABLE 1
DATA SET 1
25  23  23  27  27  31  31  25
31  27  25  25  34  31  31  29
31  31  34  29  34  31  34  31
29  31  27  31  29  29  29  31
29  31  31  31  31  29  29  31
34  34  29  29  31  31  29  29
29  31  31  36  34  34  31  34
31  31  31  31  29  31  29  29
```

After this 8X8 block is wavelet transformed, quantized and rounded off, the resultant matrix takes the form as depicted in Table 2.

```
TABLE 2
DATA SET1 AFTER WAVELET TRANSFORMATION AND QUANTIZATION
-28  0  0  0  0  0  0  0
 0  -1  0  0  1  0  0  1
-1  -1  0  1  1  -1  -1  1
 0  0  1  0  0  1  1  0
-1  0  -1  -1  -1  -1  -2  1
 1  0  -1  0  1  3  1  1
 0  0  -2  -1  -1  0  1  -1
 1  -1  -1  0  -1  -1  1  -1
```

The pixel values of an area in Andhra Pradesh are shown in Table 3.

```
TABLE 3
DATA SET 3
48  42  55  51  42  51  51
48  55  51  51  42  48  51
39  39  39  39  36  36  36
51  55  55  51  48  42  44
45  45  45  45  45  45  45
39  45  45  45  38  58  58
42  45  42  51  58  48  48
39  39  48  48  48  51  51
```

The pixel values of an 8X8 block of Fig.4 are shown in Table 3.

```
TABLE 4
DATA SET 2 AFTER WAVELET TRANSFORMATION AND QUANTIZATION
-16  0  -1  0  0  0  1  0
 0  1  1  -1  0  1  0  -1
 1  -1  -1  0  0  1  1  1
 0  0  1  0  -1  1  0  0
 0  0  -2  -1  -1  0  1  -1
-2  -1  0  0  1  -1  -1  1
-1  3  1  -1  0  -2  1  -1
 0  0  1  0  -1  -2  3  -1
```
The compressed code in byte format is

11100010 00010011 10110101 10000001
E2 13 B5 81
10001001 00010100 11101100 00101101
89 04 EC 2D
01010001 10110001 00100000 10101000
51 B1 20 A8
00111010 10000001 10110100 10110010
3A B4 B2 90

The last code contains only 4 bits and to represent it into byte format, 4 zero bits are padded. The resultant compression ratio can be determined as

\[
\text{Compression ratio} = \frac{\text{Original number of bits}}{\text{Number of bits after compression}} = \frac{512}{160} = 3.2
\]

Full Incremental Compilation runs Analysis & Synthesis, Fitter, Assembler and Timing Analyzer at once. Analysis & Synthesis analyzes design files and creates the project data base. Fitter places and routes the design. Assembler creates one or more programming files for programming or configuring the target device. Lastly, Timing Analyzer analyzes the speed performance of the implemented circuit. Table 5 shows compression ratios and time taken for full incremental compilation for two sample images.

| TABLE 5 |
| RESULTS OF COMPRESSION FOR DIFFERENT TYPES OF IMAGERY |
| Type of imagery | Block size | Compression ratio | Full compilation time |
| Monochromatic image of an area in Bangladesh | 8X8 | 3.35 | 52ms |
| Monochromatic image of an area in Andhra Pradesh | 8X8 | 3.2 | 56ms |

VI. CONCLUSION

The Wavelet Transform based image compression technology has been proved to work efficiently and yields the compression ratio around 3.3, with the minimum degradation in the data quality of the reconstructed image. The entire operation is done at clock frequencies 2MHz and 50MHz. The operation of compression for 8 lines consisting of 6000 pixels per line is required to be completed in less than 3ms. The design utilizes only 592 Logic Elements out of the available 12,000 Logic Elements on the target FPGA, which is the Cyclone EP1C12Q240C6 and consumes less than 1% of the memory space. All the logics were successfully simulated, tested and verified with the manual calculation. The paper covers only the Haar wavelets and Huffman Coding. Testing using other wavelets and implementing Arithmetic Coding will be the future activity of this paper.

REFERENCES


