© 2018 IJSRSET | Volume 4 | Issue 2 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
National Conference on Advanced Research Trends in Information and
Technologies (NCARTICT-2018), Department of IT, L. D. College of Engineering,
Ahmedabad, Gujarat, India In association with
International Journal of Scientific Research in Science, Engineering and Technology

A Survey of Efficient CCSDS Recommended DWT Decompressor

Shailja M. Maniya[1], Bakul Panchal[2]
[1]M.E. (I.T.) Student, I.T. Department, L.D. College of Engineering Gujarat Technological University, Ahmedabad, Gujarat, India
[2]Assistant Professor, I.T. Department, L.D. College of Engineering Gujarat Technological University, Ahmedabad, Gujarat, India

ABSTRACT

Now a days many satellites are being launched and data retrieval from the satellite on the earth is very expensive task as it requires too much bandwidth. So, we need compression to get the data using less bandwidth and it can also speed up file transfer. For space system there is one committee called Consultative Committee for Space Data System (CCSDS) which publishes the recommended standards on compression methodology for different type of data. CCSDS-122.0-B-1 Standard defines a compression methodology for compression of two dimensional Image Data. Inverse Discrete Wavelet Transform (IDWT) is used for the decompression of two dimensional Image Data which can be considered as time consuming part of the Decompressor. To speed up the decompressor this Inverse Discrete Wavelet Transform (IDWT) can be run parallel using different high-speed hardware. In this paper, we will survey various ways through which efficient CCSDS recommended Discrete Wavelet Decompressor can be made.
Keywords: GPGPU, CCSDS, Discrete Wavelet Transform (DWT), CUDA, NVIDIA, SPIHT, Rice decoding.

I. INTRODUCTION

The CCSDS Image Data Compression is the most widely used particularly for the grayscale image data. This algorithm is very useful for any imaging instruments application. The algorithm is designed in such a way that Its complexity remains sufficiently low so that it can be feasibly implemented on high-speed hardware.

This algorithm is intended to be used for on-board spacecrafts. Compression is used in order to save bandwidth usage & storage space required to save the image data as well as time can also be saved. On the other side, real time need is to decompress the two dimensional image data as soon as it is downlinked on the earth.

Various data Compression technique uses wavelet transformed data for better compression performance. There are basically two types of compression supported by this CCSDS standard i.e. Lossless Data Compression and Lossy Data Compression. In Lossless Data Compression, data is compressed in such a way that it can be recovered easily on decompression whereas in Lossy data compression technique, data cannot be reproduced without some distortion. Lossy Compression technique uses 9/7 Float Discrete Wavelet Transform, Lossless Compression Technique uses 9/7 Integer Discrete Wavelet Transform. In this paper, we are going to survey for the Lossless Data Decompression technique.

II. CCSDS RECOMMENDED DWT DECOMPRESSOR

In this standard, decompressor consists of two functional parts i.e. bit plane decoding compressed data followed by two dimensional inverse discrete wavelet transform of wavelet transformed data as shown in figure 1.

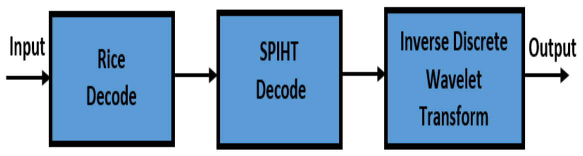

Figure 1. Overview of the Decoding system

A. Rice Decoding

Rice encoding is special case of Golomb coding which can be used to reduce the bits required to represent the lower value numbers. Rice encoding used for encoding of DC Coefficients in the CCSDS-122 standard. An algorithm for rice decoding is as shown below:
Assume Given a Constant C, any Symbol S can be represented using quotient (Q) and reminder (R), where

$$S = Q \times M + R \quad (1)$$

A. Rather than representing both Q and R as binary values, Rice encoding represents Q as a unary value and R as a binary value. Rather than representing both Q and R as binary values, Rice encoding represents Q as a unary value and R as a binary value.

B. For those not familiar with unary notation, a value N may be represented by N 1s followed by a 0.

C. Example: 4 = 11110 and 2 = 110.

If given Bit Length K, Compute the Modulus M, using by given equation (2)

$$M = 2^K \qquad (2)$$

Then perform the following steps for each encoded Symbol (S),

1. Determine Q by counting the number of 1s before the first 0.
2. Determine R reading the next K bits as a binary value.
3. Write out S as Q x M + R.

Example: decode the encoded value 100010 when K = (M=16)

1. Q=1
2. R = 0b0010 = 2
3. S = Q x M + R=1 x 16 + 2 = 18

B. SPIHT Decoding

SPIHT (Set Partitioning in Hierarchical Trees) is well known algorithm to encode the image using bit planes and it performs two passes for each bit-plane. While one pass computes sign values and the implicit location information of significant wavelet coefficients, the second pass sends the refined bit values of the significant coefficients which are determined up to the current bit plane.

The decoder rebuilds the wavelet coefficients according to the importance of the information. This process is serial in nature because the importance of information will decide the position of the next node. Thus, decoding process can only finish when all nodes are decoded. Therefore, it is clearly said that SPIHT process should be performed on the CPU.

C. Inverse Discrete Wavelet Transform

As explained in section 1, we are going to do survey for the lossless data compression we have to use Inverse 9/7 Integer Discrete Wavelet Transform in decompressor. Inverse Integer DWT mapping is from two set of wavelet coefficients C_j and D_j back to the signal vector x_i as mentioned in [1] which is given by equations (1) to (6). Here we require special boundary filters to get the data back.

In equation given below, we first compute the even indexed signal values i.e. $x_0, x_2, \dots, x_{2N-2}$ using DWT coefficients. After computation of these even indexed vector values, odd indexed values can be calculated i.e.$x_1, x_3, \dots, x_{2N-1}$.

$$x_1 = D_0 + \left\lfloor \frac{9}{16}(x_0+x_2) - \frac{1}{16}(x_2+x_4) + \frac{1}{2} \right\rfloor \qquad (1)$$

$$x_{2j+1} = D_j + \left\lfloor \frac{9}{16}(a+b) - \frac{1}{16}(c+d) + \frac{1}{2} \right\rfloor \qquad (2)$$

where a= x_{2j} , b= x_{2j+2} , c= x_{2j+2}, d= x_{2j+4} for j=1,…,N-3

$$x_{2N-3} = D_{N-2} + \left\lfloor \frac{9}{16}(e+f) - \frac{1}{16}(g+f) + \frac{1}{2} \right\rfloor \qquad (3)$$

where e= x_{2N-4},f= x_{2N-2}, g= x_{2N-6}

$$x_{2N-1} = D_{N-1} + \left\lfloor \frac{9}{8} x_{2N-2} - \frac{1}{8} x_{2N-4} + \frac{1}{2} \right\rfloor \qquad (4)$$

$$x_0 = C_0 + \left\lfloor -\frac{D0}{2} + \frac{1}{2} \right\rfloor \qquad (5)$$

$$x_{2j} = C_j + \left\lfloor -\frac{D_{j-1}+D_j}{4} + \frac{1}{2} \right\rfloor \quad \text{for } j = 1,\ldots, N-1 \qquad (6)$$

D. Two dimensional Single Level Inverse DWT

The single-level 2-d DWT transform[1] shall be inverted by repeated application of the 1-d inverse to columns and rows of the transformed data array in the reverse order to that in which the 1-d transforms were applied:

a) each column shall be inverted to produce the intermediate transformed data arrays:

1) The 1-d DWT inverse shall be applied to columns of the LL and LH subbands to obtain the intermediate horizontal low-pass array of figure 3(b),
2) The 1-d DWT inverse shall be applied to columns of the HL and HH subbands to obtain the intermediate horizontal high-pass array of figure 3 (b);

b) the 1-d DWT inverse shall be applied to rows of the intermediate horizontal low-pass and horizontal high-pass arrays to recover the original image array.

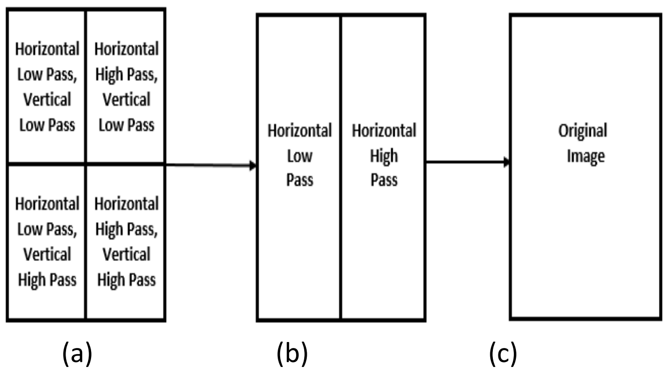
Horizontal Low Pass, Vertical Low Pass
Horizontal High Pass, Vertical Low Pass
Horizontal Low Pass, Vertical High Pass
Horizontal High Pass, Vertical High Pass
Horizontal Low Pass
Horizontal High Pass
Original Image
(a)
(b)
(c)

Figure 3. One level 2d Inverse DWT

E. Two dimensional three level Inverse DWT

The inversion process of a multi-level DWT shall be as follows [1]:

a) The four subbands of highest level, LL3, LH3, HL3, HH3, shall be inverted using an inverse single-level 2-d DWT to yield the single subband LL2, which then replaces the higher-level subbands in the transform data matrix;

b) The four subbands LL2, LH2, HL2 and HH2 shall be inverted to yield the single subband LL1, which again replaces the higher-level subbands in the transform data matrix;

c) A final single-level 2-d inverse DWT shall be applied to subbands LL1, LH1, HL1 and HH1 to reproduce the original image.

III. LITERATURE REVIEW

A. Wavelet based decoding system [2]

Changhe Song, Yunsong Li, and Bormin Huang at el. [2] has implemented the decoding system for satellite images. They have implemented a wavelet based decoding system which contains SPIHT with Reed-Solomon decoding. Overview of the decoding system is as shown in below fig. 4.

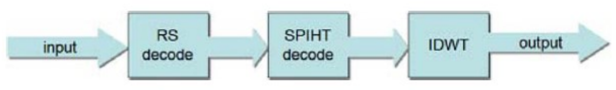

Figure 4. Overview of decoding system [2]

In this paper, they have used float 9/7 inverse DWT as a wavelet transform with SPIHT and Reed-Solomon decoding.

Table 1.
CPU TIME OF EACH COMPONENT OF THE DECODING SYSTEM

RS decoding	18 ms
SPIHT decoding	62 ms
IDWT	841 ms

As shown in table 1, IDWT is the most time consuming process of the decoding system. They have used General Purpose Graphics Processing Unit (GPGPU) for the faster computation of IDWT. They have also used shared memory for the better time performance of the IDWT computation. GPU based IDWT model is as shown in fig. 5.

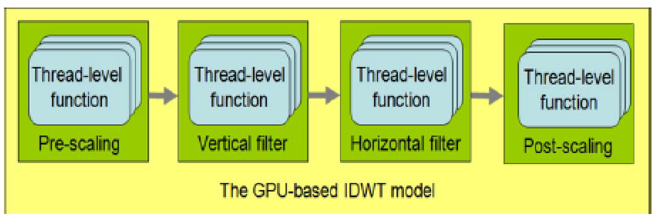

Figure 5. IDWT model on GPU [2]

Fetching steps of horizontal filter and vertical filer are shown in figure 6 & 7 respectively.

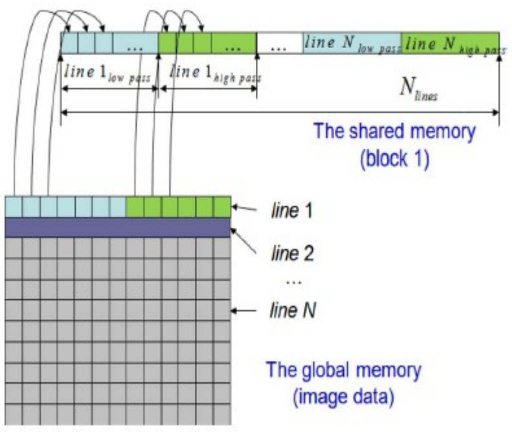

Figure 6. the fetch step of horizontal filter [2]

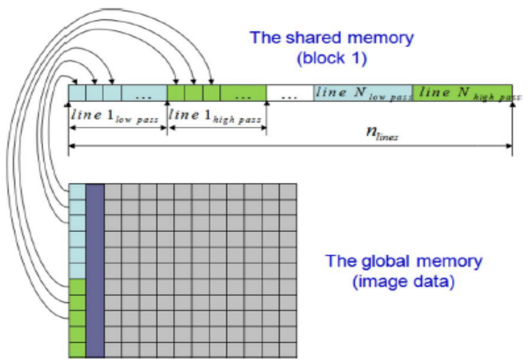

Figure 7. the fetch step of vertical filter [2]

Thus they have prepared CPU-GPU pipelined fashion decoding system as shown in figure 8.

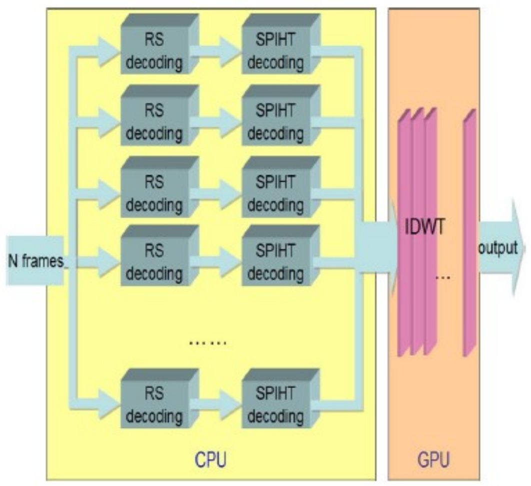

Figure 8. Overview of RS+SPIHT+IDWT decoding system [2]

B. Time efficiency Comparison of Wavelet transform and its inverse on different platforms [3]

Abhishek S. Shetty, Abhijit V. Chitre and Yogesh H. Dandawate at el [3] has done the comparison for the wavelet and Inverse wavelet transform on different platforms. They have used 9/7 Integer Discrete wavelet transform and its inverse for this comparison. Three different platforms are used which are MATLAB, Python using OpenCV and Python using PIL (Python Imaging Library).

As a result the comparison chart is prepared as shown in figure 9. It is observed that Pthon-OpenCV gives better result in terms of time and it is also open source. These results are being obtained using image size starting from 512x512 to 6000x6000.

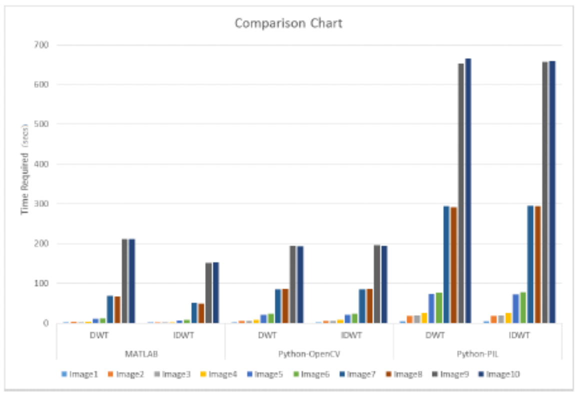

Figure 9. Comparison Chart [3]

C. Efficient parallelization of Discrete Wavelet Transform [4]

Anastasis Keliris, Vasilis Dimitsasy, Olympia Kremmyday, Dimitris Gizopoulosy and Michail Maniatakosz at el [4] has implemented the Discrete wavelet transform on multi core cpu, Many Integrated Cores (MIC) cpu as well as on NVIDIA GPU. In this paper, 9/7 integer DWT is used as a wavelet transform.. They have used the four evaluation systems as shown in table 2.

Table 2.

CONFIGURATION OF THE EVALUATION SYSTEMS

	System 1	System 2
CPU	AMD Phenom II X4 965 4 cores @3.4 GHz 45nm, released: 2009	Intel Xeon E5-2680 8 cores @2.7GHz 32nm, released: 2012
Accelerator	Nvidia Tesla C2070 (Fermi) 448 CUDA cores 40nm, released: 2010	Intel Xeon Phi 5110P 60 cores @1.053GHz 22nm, released: 2012

They have used the algorithm for the transpose of the matrix because this operation is memory-intensive operation. Algorithm [4] for the matrix transpose is as written below:

```
Algorithm 1: Cache-oblivious mxn recursive matrix
transposition
1 in, out: input/output image
2 r off, c off: rows/columns offset
3 Define: BLOCKSIZE (architecture dependent)
4 Function trans (in, out, r off, c off, m, n)
5 if m > BLOCKSIZE or n > BLOCKSIZE
then
6 if n > m then
7 nhalf  n/2;
8 trans (in, out, r off, c off+nhalf, m, nhalf);
9 else
10 mhalf  m/2;
11 trans (in, out, r off+mhalf, c off, mhalf, n);
12 end
13 else
14 rlimit   r off + n;
15 climit   c off + m;
16 for i   r off to rlimit do
17 for j   c off to climit do
18 out[j][i] = in[i][j];
19 end
20 end
21 end
22 return
```

D. Energy consumption analysis of CCSDS image compression running on two different platforms [8]

Christofer Schwartz, Marcelo S. pinho, at el [8] has implemented CCSDS-122 DWT based compressor on the CPU as well as on the GPU. CPU specifications are : Intel Core i7 - 3610QM (third generation) with 4 cores (8 threads) working at a frequency of 2; 241; 003 KHz and GPU specifications are NVIDIA GeForce GT 630M (2.1 Streaming Multiprocessor Capability) — this GPU has two multiprocessor (MP) with 48 cores each (total of 96 cores) working at a clock of 950; 000 KHz.

In this paper, they have implemented bit-plane encoder on the Host side (CPU) and DWT is performed on device (GPU). Timing given by the host and host + device system are analysed in this paper.

IV. COMPARISON

As described in Section III, Changhe Song at el [2] has shown in their results that IDWT part of the decoding system is the most time consuming part in table 1. As per the result of [2], CPU-GPU pipeline system gives better performance than IDWT on GPU and SPIHT + Reed-solomon on CPU as shown in table 3 & 4.

Table 3.

EXECUTION TIME OF THE IMPROVED SYSTEM WITH IDWT ON GPU AND THE 6-FRAME CONCURRENT RS AND SPIHT DECODING CPUS

	Original (ms)	Improved (ms)	Speedup
RS	18	3.7	5
SPIHT	62	12.9	5
IDWT	841	13.3	63
Total	921	29.9	**31**

Table 4.

EXECUTION TIME WITH THE CPU-GPU PIPELINING

	Time (ms)	Speedup
CPU (RS+SPIHT decoding)	16.6	5
GPU (IDWT)	13.3	63
Total	16.6	**55**

As per the result of [3], Python-OpenCV gives better timing results for wavelet transform, further use of GPGPU can still reduce time.

Paper [4] has given results for the DWT algorithm time for different size of images with different types of platforms which is shown in table 5. As shown in the table 5, NVIDIA Tesla gives the best performance from all of the different platforms.

Table 5.

EXECUTION TIMES (IN MS) OF THE DWT ALGORITHM

	256x256	512x512	1024x1024	2048x2048	4096x4096	8192x8192
Phenom ser.	5.7	11.7	66.4	303.4	1348.6	7809.8
Phenom opt.	1.9	4.5	20.4	55.4	197.6	740.6
Tesla	0.2	0.5	1.2	3.9	14.6	58.4
Xeon E5 ser.	1.0	3.6	24.6	108.2	546.3	2589.6
Xeon E5 opt.	1.2	2.3	6.4	16.3	47.2	144.8
Xeon Phi	7.8	28.8	60.1	130.2	263.7	590.1

In terms of energy consumption, the most suitable hardware for the CCSDS-122 Compression algorithm has been experimented in [8]. The result is shown in fig. 10 using rate-distortion-cost curve.

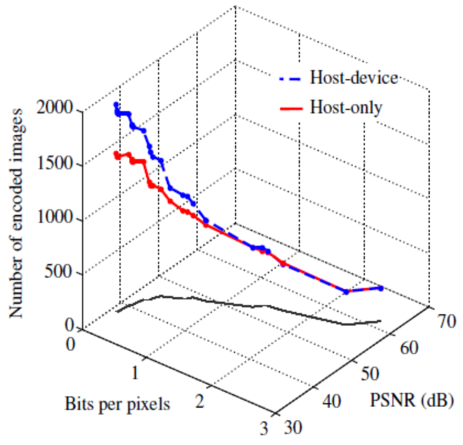

Figure 10. rate-distortion-cost curve [8]

V. CONCLUSION

Different ways to make efficient CCSDS-122 based Decompresssor has been viewed in this paper. This standard Decompressor contains two modules, IDWT and Bit plane Decoder, from which IDWT is most time consuming part of this decoding system. Different ways to do the computation of IDWT in parallel are reviewed, in which NVIDIA, GPU gives best results in terms of time as well as in terms of energy consumption, too. Moreover, CPU-GPU pipeline decoding system can be used to make efficient CCSDS recommended DWT Decompressor for better performance.

VI. REFERENCES

[1]. "Image Data Compression", Recommendation for space data system standards, CCSDS 121.0-B-1. Blue Book,November 2005.

[2]. Changhe Song, Yunsong Li, and Bormin Huang,"A GPU-Accelerated Wavelet Decompression System With SPIHT and Reed-Solomon Decoding for Satellite Images",IEEE journal of selected topics in applied earth observations and remote sensing, vol. 4, no. 3, september 2011.

[3]. Abhishek S. Shetty, Abhijit V. Chitre and Yogesh H. Dandawate, "Time Efficiency Comparison of Wavelet and Inverse Wavelet Transform on Different Platforms",International Conference on Computing Communication Control and automation (ICCUBEA) IEEE-2016.

[4]. Anastasis Keliris, Vasilis Dimitsasy, Olympia Kremmyday, Dimitris Gizopoulosy and Michail Maniatakosz, " Efficient parallelization of the Discrete Wavelet Transform algorithm using memory-oblivious optimizations", 25th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS),pp.25-32, 2015

[5]. John Nickolls,"GPU Parallel Computing Architecture and CUDA Programming Model " , Hot chips 19 Symposium(HCS) IEEE,pp.1-12, 2007

[6]. Khoirudin and Jiang Shun-Liang ,"Gpu application in cuda memory", Advanced Computing: An International Journal (ACIJ), Vol.6, No.2,pp.1-10, March 2015

[7]. NVIDIA, " NVIDIA CUDA Compute Unified Device Architecture", United States, 2007.

[8]. Christofer Schwartz, Marcelo S. pinho,"an energy consumption analysis of ccsds image compressor running in two different platforms", IGARSS-IEEE, pp. 1640-1650,2014.

[9]. William A. Pearlman,Amir Said," Set Partition Coding: Part I of Set Partition Coding and Image Wavelet Coding Systems",foundations and trend in signal processing,Vol. 2,No.2, pp.97-180,2008.

[10]. William A. Pearlman,Amir Said," Image Wavelet Coding Systems: Part II of Set Partition Coding and Image Wavelet Coding Systems",foundations and trend in signal processing,Vol. 2,No.3, pp.181-246,2008.

[11]. William A. Pearlman,Amir Said," A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees", ieee transactions on circuits and systems for video technology, vol. 6, no. 3,pp.243-249, june 1996

[12]. Jay W. Schwartz, Richard C. Barker," Bit-Plane Encoding: A Technique for Source Encoding", IEEE transactions on aerospace and electronic systems, vol. aes-2, no. 4, pp.385-392, july 1966.

[13].