NASA Conference Publication 3277

1994 Science Information Management and Data Compression Workshop

Proceedings of a workshop sponsored by the
National Aeronautics and Space Administration
in Cooperation with the
Institute of Electrical and Electronics Engineers
Geoscience and Remote Sensing Society
held at the NASA Goddard Space Flight Center
Greenbelt, Maryland
September 26-27, 1994

NASA Conference Publication 3277

1994 Science Information Management and Data Compression Workshop

Edited by
James C. Tilton
NASA Goddard Space Flight Center
Greenbelt, Maryland

Proceedings of a workshop sponsored by the
National Aeronautics and Space Administration
in Cooperation with the
Institute of Electrical and Electronics Engineers
Geoscience and Remote Sensing Society
held at the NASA Goddard Space Flight Center
Greenbelt, Maryland
September 26-27, 1994

National Aeronautics
and Space Administration

Goddard Space Flight Center
Greenbelt, Maryland 20771

1994

This publication is available from the NASA Center for AeroSpace Information,
800 Elkridge Landing Road, Linthicum Heights, MD 21090-2934, (301) 621-0390.

FOREWORD

The Science Information Management and Data Compression Workshop was held on September 26-27, 1994, at the NASA Goddard Space Flight Center, Greenbelt, Maryland. This NASA Conference Publication serves as the proceedings for the workshop. The workshop organized by the Information Sciences Technology Branch, Space Data and Computing Division of the NASA Goddard Space Flight Center, and was supported by the Office of Advanced Concepts and Technology, NASA Headquarters. The workshop was held in cooperation with the Institute of Electrical and Electronics Engineers (IEEE) Geoscience and Remote Sensing Society.

The goal of the Science Information Management and Data Compression Workshop was to explore promising computational approaches for handling the collection, ingestion, archival and retrieval of large quantities of data in future Earth and space science missions. It consisted of eleven presentations covering a range of information management and data compression approaches that are being or have been integrated into actual or prototypical Earth or space science data information systems, or that hold promise for such an application.

Papers were selected from papers submitted in response to a widely distributed Call for Papers. Eleven papers were presented in 3 sessions. Discussion was encouraged by scheduling ample time for each paper.

The workshop was organized by James C. Tilton and Robert F. Cromp of the NASA Goddard Space Flight Center.

Workshop Organizers

James C. Tilton
Mail Code 935
NASA GSFC
Greenbelt, MD 20771
phone: (301) 286-9510
FAX: (301) 286-1776
Internet:
tilton@chrpisis.gsfc.nasa.gov

Robert F. Cromp
Mail Code 935
NASA GSFC
Greenblet, MD 20771
phone: (301) 286-4351
FAX: (301) 286-1776
Internet:
cromp@sauquoit.gsfc.nasa.gov

CONTENTS

PAPERS

Performance of Customized DCT Quantization Tables on Scientific Data*

Viresh Ratnakar
University of Wisconsin-Madison
Computer Science Department
Madison, WI

Miron Livny
University of Wisconsin-Madison
Computer Science Department
Madison, WI

Abstract

We show that it is desirable to use data-specific or *customized* quantization tables for scaling the spatial frequency coefficients obtained using the Discrete Cosine Transform (DCT). DCT is widely used for image and video compression [MP89, PM93] but applications typically use default quantization matrices. Using actual scientific data gathered from divers sources such as spacecrafts and electron-microscopes, we show that the default compression/quality tradeoffs can be significantly improved upon by using customized tables. We also show that significant improvements are possible for the standard test images *Lena* and *Baboon*. This work is part of an effort to develop a practical scheme for optimizing quantization matrices for any given image or video stream, under any given quality or compression constraints.

1 Introduction

We are developing an environment for "production-mode" compression of still-image and video data, where the user can specify constraints on the desired quality and compression ratio, and the compressor produces the best results under those constraints without any human assistance. Under both the JPEG and MPEG compression standards, quality and compression-ratio can be varied by varying the DCT coefficients' quantization table. Most existing encoders simply use a default table and scale it up or down by a small factor to achieve different qualities/compression-ratios. This paper shows that customized quantization tables can outperform scaled default tables to a high degree. We are exploring efficient algorithms for designing these customized tables.

The test images and video streams used for the performance study were some spacecraft images (*Earth*, *Venus*), some molecular-biology images (*Cell*, *Egg*), and some standard images

*This work was partially supported by NASA grant NAGW-3914 and NSF grant IRI-9224741.

used in image compression literature (*Lena*, *Baboon*). In every case, substantial gains were obtained. At a given bit rate, Peak Signal-to-Noise Ratio (PSNR) could usually be improved by about 1-2 dB, while at a given PSNR, bit rate could be reduced by about 0.2 bits per pixel (bpp).

The rest of this paper is organized as follows: Section 2 outlines the use of DCT for image compression and the role of quantization. Section 3 presents the performance of customized quantization tables. Subsection 3.1 shows the results for two standard images, *Lena*, and *Baboon*. Subsection 3.2 shows the results for four scientific data streams. Conclusions are presented in section 4.

2 Discrete Cosine Transform

JPEG and MPEG work by dividing each image (or frame) into blocks of size 8×8 and transforming each block using the Discrete Cosine Transform (DCT). This transformation results in an 8×8 block F of 64 coefficients for each image block f. These coefficients define the unique representation of f as a linear combination of 64 predefined *basis blocks* of the DCT.

The basis blocks capture different spatial frequencies of an image. $F(0,0)$ is the coefficient of the term with zero spatial frequency (the DC component) while $F(7,7)$ is the coefficient of the term with highest frequencies in the x and y directions. Details can be found in [RY90].

The advantage in using DCT is that it compacts most of the signal energy into the low frequency coefficients. The human eye is relatively insensitive to high spatial frequencies which can hence be ignored. In fact, the entire block F of coefficients is quantized using some 8×8 quantization table Q. Thus, the value $F(u,v)$ is stored as the integer closest to $\frac{F(u,v)}{Q(u,v)}$. Higher frequencies are quantized more coarsely (i.e. with a greater value of $Q(u,v)$) than lower frequencies. A good proportion of the high-frequency coefficients get quantized to zero. This enables the block to be compressed efficiently using entropy coding techniques such as Huffman coding or arithmetic coding [Jai89, PM93]. However, this compression is lossy as the reproduced values of the coefficients will not necessarily be the same as the original values.

The quantization table Q ultimately determines the compression ratio and the quality of the reconstructed image. JPEG allows only a fixed table for the entire image. Most encoders set this table to the table suggested in the standard [PM93]. For better quality, the entire table is scaled down while for higher compression, the entire table is scaled up by a small factor which is called the *qscale*.

MPEG allows the quantization table to be changed along a video stream. In addition, the table used may be scaled up or down by multiplying by a *qscale* on a per-macroblock basis. This scaling is done by heuristically determining regions of low activity and high activity and adjusting *qscale* accordingly (See, for example, [CP84]). The table used is generally the one suggested in the standard [MP89].

3 Performance of customized quantization tables

We used Peak Signal-to-Noise Ratio as the measure of quality of a decompressed image. If I is an $M \times N$ grayscale image with pixel values in the range [0..255], and I is approximated by image I', then

$$\text{PSNR} = 10.0 * \log_{10}\left(\frac{\sum_{i,j}(255)^2}{\sum_{i,j}(I(i,j) - I'(i,j))^2}\right).$$

An approximation with PSNR greater than about 37.0 dB is usually indistinguishable from the original image, to the human eye.

Degree of compression was measured as bits per pixel (bpp) used. For the 8-bit grayscale images used, compression ratio is equal to (8/bpp): 1.

For each test image or video stream, we plotted PSNR vs bpp for customized tables and default tables. To obtain customized tables, for every image/stream at every PSNR we searched a wide range of quantization tables to find the best performance (in terms of actual bit rate), using trial and error. The default tables were obtained by multiplying the standard tables suggested by JPEG (for still images) and MPEG (for video streams) by a *qscale* in the range $\frac{1}{8} \cdots \frac{31}{8}$, as is allowed under the two standards [MP89, PM93].

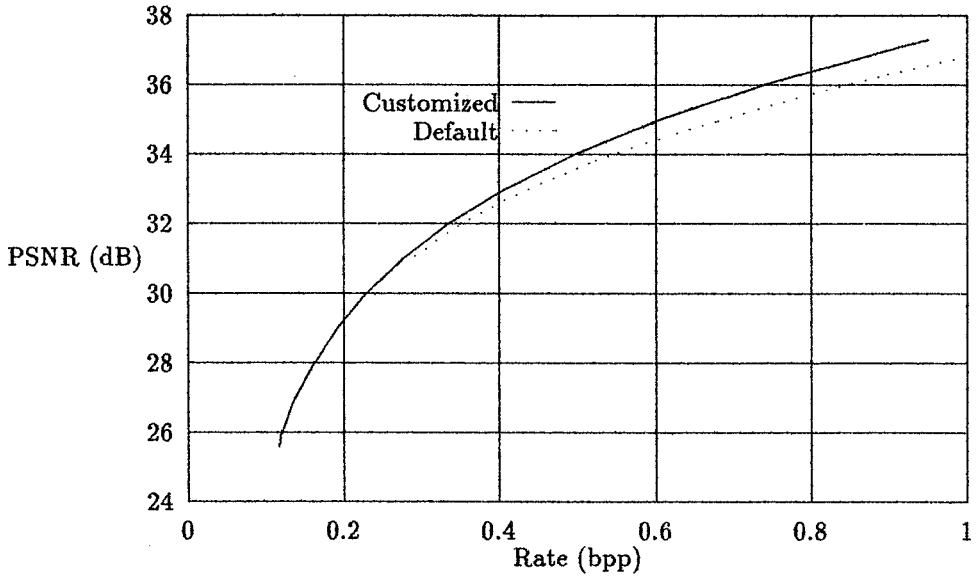


Figure 1: Quality vs Rate curves for *Lena*

The range of the plots was chosen as 0..1 bpp. The lowest bit rate plotted for the default curve was the one achieved by multiplying the default table by a *qscale* of 31/8. For the customized curve, the lowest bit rate was that achieved by maximizing all table entries. The highest bit rate for the default curve (*qscale* = 1/8) was typically in the 0.9 to 1.2 bpp range. All the plots go up to 1 bpp for ease of comparison. The images are not reproduced here because of space constraints.

3.1 Performance results for *Lena* and *Baboon*

Figures 1 and 2 show the results for the standard 512 × 512 8-bit grayscale images *Lena* and *Baboon*, respectively. The default table used was that suggested by the JPEG standard [PM93]. The advantage of using customized tables is seen to be more at higher rates and better qualities. For example, for *Lena*, an improvement of about 0.5 dB in quality can be obtained at rate of 0.8 bpp. The improvement is more pronounced for *Baboon* which

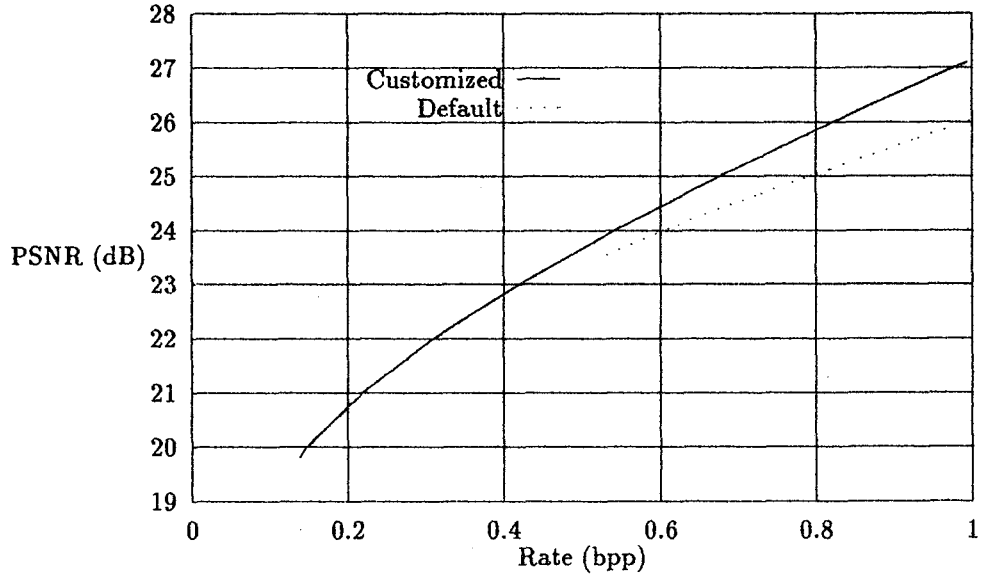


Figure 2: Quality vs Rate curves for *Baboon*

offers a gain of about 1 dB at 0.8 bpp. Comparing bit rates at fixed qualities, we see that a reduction by about 0.1 bpp is achieved at 35 dB for *Lena*. Once again, the difference is more pronounced for *Baboon* where, for example, a reduction of nearly 0.2 bpp can be seen at about 26 dB. It is interesting to note that *Baboon* has a lot of high frequency content and is known to be hard to compress. In general, the improvements offered by customized tables were greater for hard-to-compress images. For such images, the default tables were not able to quantize the high frequencies efficiently, while the customized tables did a much better job. This was seen from the fact that the variances of quantized high-frequency coefficients were lower with customized tables than with default tables.

3.2 Performance results for scientific data

All the results in this section are based on video streams compressed using the I-frames of MPEG-I. Figure 3 shows the results for a stream of 320 × 300 pictures of Earth taken from a satellite. The original pictures were in raw 8-bit grayscale format. Figure 4 shows the results for a stream of 480 × 480 pictures of Venus shot by a NASA spacecraft. Again, the

original pictures were in raw 8-bit grayscale format. Figure 5 plots the results for a stream of 352×240 8-bit grayscale pictures of a cell. Finally, Figure 6 refers to a computer generated sequence of 320×240 8-bit grayscale images. For *Earth*, quality improvements went up

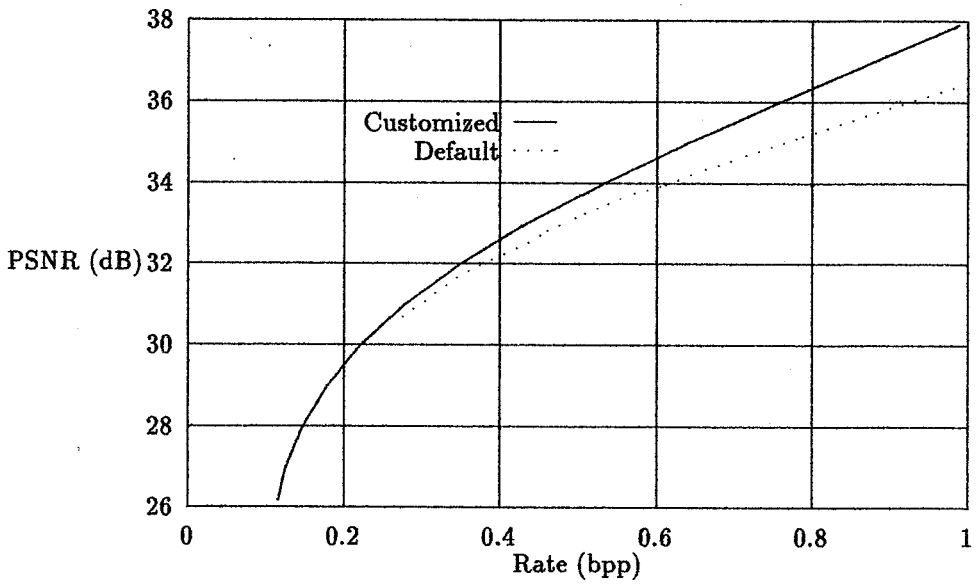


Figure 3: Quality vs Rate curves for *Earth*

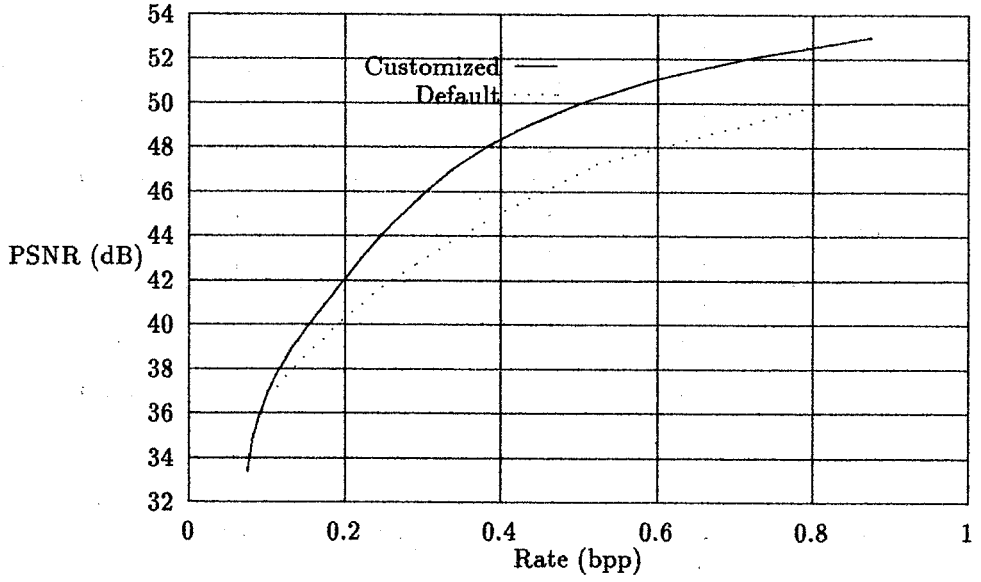


Figure 4: Quality vs Rate curves for *Venus*

to about 2 dB while bit rate reductions up to 0.2 bpp were obtained. *Venus* was rather easy

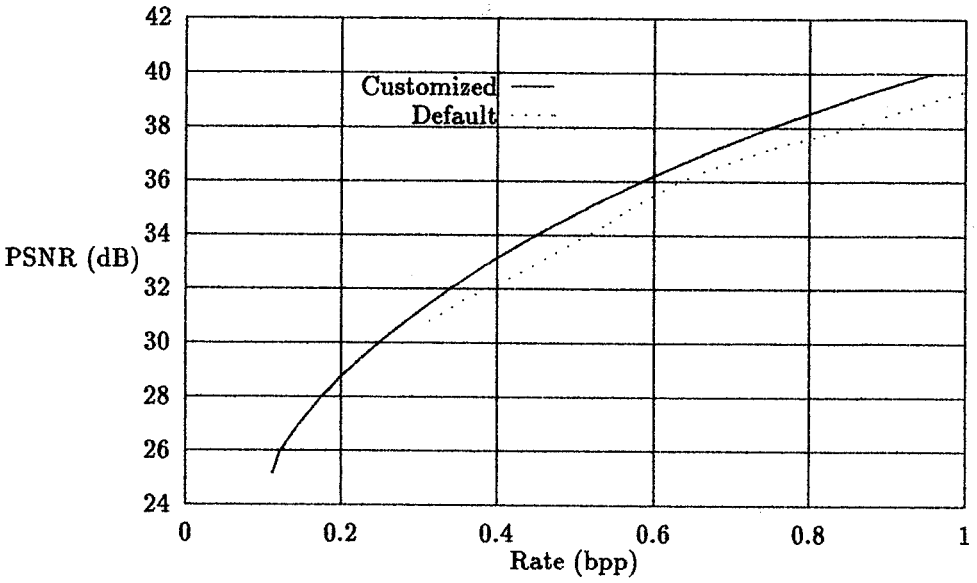


Figure 5: Quality vs Rate curves for *Cell*

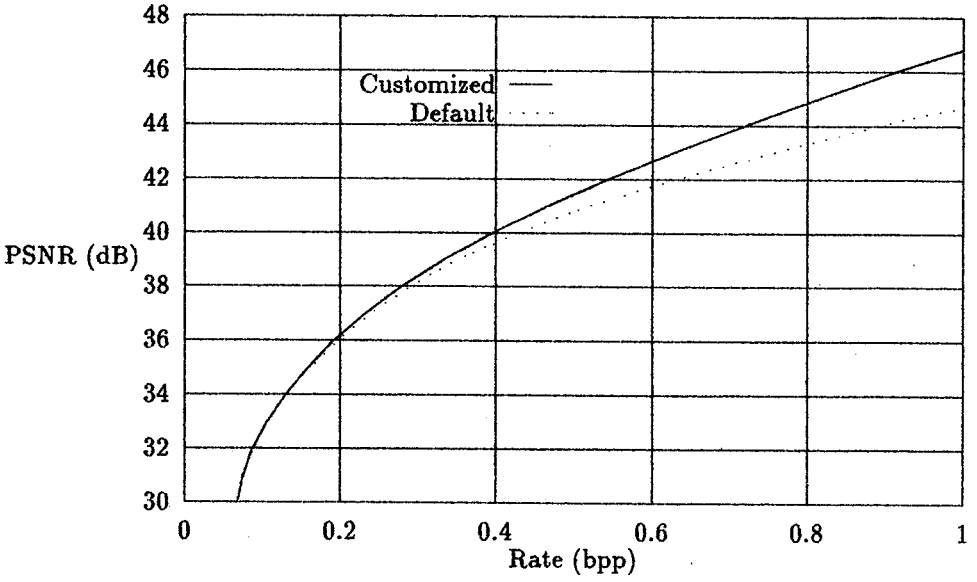


Figure 6: Quality vs Rate curves for *Egg*

to compress as can be seen from the fact a PSNR of 42 dB is achievable at merely 0.2 bpp. The plot for *Venus* shows that the improvement in quality varied between 2 dB (at 0.2 bpp) and 3 dB (at 0.6 bpp). The reduction in rate varied between 0.08 bpp (at 42 dB) and 0.3 bpp (at 50 dB).

For *Cell*, gains in quality were about 1 dB at every bit rate, while reduction in bit rate varied between 0.05 (at 32 dB) to about 0.1 bpp (at 39 dB). The last stream, *Egg*, displayed gains in PSNR up to 2 dB (at 0.9 bpp), and rate reduction up to 0.2 bpp (at 45 dB). The PSNR for *Egg* was better than that for *Cell* at every bit rate. This is to be expected as the stream *Egg* was generated using computer animation and had lesser high-frequency content.

We can see that customized quantization tables improved quality and compression-ratio for every image and video stream. The improvements varied a bit in amount across different images and streams, but were usually substantial enough to justify the use of customized tables, especially at bit rates exceeding 0.6 bpp.

4 Conclusion

Using image data gathered from widely different sources, we have shown that the performance of default tables can always be significantly improved upon. A reduction of 0.2 bpp for 1000 pictures of earth, each 320×300 8-bit grayscale, translates to an additional saving of around 2.4 Megabytes.

We are developing algorithms to design customized quantization tables efficiently, to exploit these possible savings in bit rate and gains in quality. A good choice of the quantization table Q becomes extremely important for production-mode compression environments. In production-mode, the compressor might be presented with widely varying image- and stream-types. A naive choice of Q might give poor performance, as we have seen (particularly for images with large high-frequency content).

For both the default and customized cases we have shown the quality/compression tradeoffs. But deciding which point on the curve to choose, given some constraints (such as exact values or ranges of tolerance for rate and quality), is also a non-trivial problem.

We also tried to exploit customized tables further by adaptively scaling *qscale* on a per-macroblock basis. This did not yield any improvement in PSNR in most cases. However, adaptive scaling does offer better *visual* quality. Further work is needed to detect and exploit scene changes. A new customized table should be introduced on a scene change. Further gains can also be obtained by similarly customizing quantization tables for the non-intracoded frames of MPEG with motion compensation.

Acknowledgements

We thank researchers in the departments of Space Sciences and Molecular Biology at University of Wisconsin-Madison for providing us with video data.

References

[CP84] Chen, W. and Pratt, W. K. Scene adaptive coder. *IEEE Trans. Comm.*, 32:225–232, 1984.

[Jai89] Jain, A. K. *Fundamentals of Digital Image Processing.* Prentice Hall, Englewood Cliffs, NJ, 1989.

[MP89] MPEG proposal package description, July 1989. Document ISO/WG8/MPEG/89–128.

[PM93] Pennebaker, W. B. and Mitchell, J. L. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, New York, 1993.

[RY90] Rao, K. R. and Yip, P. *Discrete Cosine Transform: Algorithms, Advantages, Applications.* Academic Press, Inc, San Diego, California, 1990.

COMPARISON OF TRANSFORM CODING METHODS WITH AN OPTIMAL PREDICTOR FOR THE DATA COMPRESSION OF DIGITAL ELEVATION MODELS

M. Lewis.
Department of Mathematics and Computing, The University of Glamorgan,
Pontypridd, Mid-Glamorgan, CF37 1DL., U.K.

Tel.U.K. 443 480480 Extn. 2714, Fax: U.K. 443 482711.
Email : MLEWIS@uk.ac.glam

ABSTRACT

Statistical encoding techniques enable the reduction of the number of bits required to encode a set of symbols, and are derived from their probabilities. Huffman encoding [1] is an example of statistical encoding that has been used for error-free data compression. The degree of compression given by Huffman encoding in this application can be improved by the use of prediction methods. These replace the set of elevations by a set of corrections that have a more advantageous probability distribution. In particular, the method of Lagrange Multipliers for minimisation of the mean square error has been applied to local geometrical predictors [3]. Using this technique, an 8-point predictor achieved about a 7% improvement over an existing simple triangular predictor [2].

In this paper, comparisons have been made between this predictive encoding methods and a transform coding technique, the Two-Dimensional Discrete Cosine Transform (2D-DCT). Transform coding allows greater compression but is computationally intensive and is subject to a greater degree of error on reconstruction of the data The Discrete Cosine Transform coding method can be combined with either Huffman encoding or Run Length Encoding (RLE) of the DCT coefficients to achieve greater compression. The method of blocking the DCT coefficients before Huffman encoding gives a better performance than Run Length encoding of the DCT coefficients The best compression achievable for the same data set using the slow DCT algorithm with blocking is about 35.24:1, i.e. a storage saving of 96.49% for at most an error of 5 metres and a root mean square error (rmse) of 0.5. For error-free compression (accurate to the nearest metre), the simple prediction method [2] gives a compression ratio of 13.04:1 with blocking the prediction errors before Huffman encoding. This gives a storage saving of about 92.30%. Similar results for a second more variable data set give a compression ratio of 17.60:1 or a storage saving of 94.12%, again for an equivalent maximum error of about 5 metres and an rmse of about 0.8. In the error-free Huffman method with blocking, equivalent results for the second data set are a compression ratio of about 7.13:1 and a storage saving of 85.93%. The Lagrange Multiplier method [3] will give an improvement of about 7% to the error-free compression ratios quoted. Since both these algorithms are computationally expensive, a trade-off between

maximum compression ratio and speed of compression/decompression must be made.

The use of another transform technique, the two dimensional Daubechies Wavelet transforms [4,5] shows similar performances with further blocking of the coefficients before Huffman encoding. The best performance for the first data set was a storage saving of 96.04% or a compression ratio of about 30:1 with the 12-coefficient ('smooth') transform with at most an error of 3 metres and a root mean square error (rmse) of 0.5. For the second more variable data set, a storage saving of 92.33% or a compression ratio of about 13:1 with the 4-coefficient ('local') transform. Here the maximum error is 3 metres with an rmse of 0.8. Further evaluation with other wavelet transforms is being studied as well as improved preprocessing of data to improve predictor efficiencies.

1. Data Compression in Digital Terrain Models.

The main emphasis in this work has not been on the encoding methods themselves but on the prediction methods specific to terrain that allow the coding methods to work better. Two transformation methods and one statistical encoding method have been chosen to apply to DEM's. These methods have been chosen because they have the potential to form robust data compression schemes with both good compression performance and moderate to good computational requirements.

In general, an estimate of the maximum amount of compression achievable in an error-free encoding process can be made by dividing the average number of bits needed to represent each terrain height in the original source data by a first-order estimate of the entropy of the prediction error data. Since there is in general a large degree of redundancy in the source data, the prediction process enables a reduction in the entropy value through this mapping process due to the probability density function of the prediction errors being highly peaked at zero and characterised by a relatively small variance.

In order to reduce the overall amount of data needed for storage, a prediction algorithm is employed. Since there is a close correlation between adjacent height values in a DTM, the differences between the actual and predicted values can be represented by fewer bits than the original data. These differences between the predictions and the actual elevations are recorded and Huffman encoded. Some base elevations such as two known axes of elevations are also stored.

One approach to the error-free compression of digital elevation data (DEM) involves the use of an identical predictor for both encoding and decoding processes. A terrain surface is normally considered to be a two-dimensional array representation of height values. Another approach to the design of an optimum predictor proved this triangular predictor to be sub-optimal and a better predictor

was devised using the method of Lagrange Multipliers [3]. ST06 and ST08 are two tiles taken from the Ordnance Survey regular 401x401 square grid with terrain heights accurate to the nearest metre. The former tile contains sea, coastal cliffs and relatively smooth changes in contours whilst the latter contains rougher terrain with deep valleys with large changes in contour values. A small improvement can be made to the simple triangular predictor method for both the three-point and eight-point predictors by the minimisation method of Lagrange multipliers. For many data sets compression ratios above 4 or 5 are easily achievable using a error-free Huffman encoding algorithm with minor modification to the code given in [2].

2.0 Application of a Transformation Method to a Digital Elevation Model.

2.1. Performance of the Two Dimensional Discrete Cosine Transform (2D-DCT).

The prediction/Huffman method described above is suitable for error-free encoding. The DCT method normally gives some error even without explicit quantisation. This is due to the representation of the real coefficients as integers. If the method were modified to make it error-free, it would give no compression. As well as evaluating various DCT methods, it is interesting to compare the DCT and prediction methods when error-free compression is not required. The prediction/Huffman method can be used in a lossy way whereby elevations are grouped into bands. e.g.

Band	Elevations (m)	Representative Elevation (m)
0	0	0.0
1	1 2 3 4 5	3.0
2	6 7 8 9 10	8.0
3	11 12 13 14 15	13.0

with a maximum error of 2.0 metres.

The bands are then used for prediction and correction. The maximum errors are small but the root mean square error (rmse) may be relatively large as most of the elevations may be in error. Note is also made that in some of the results in this section, the elevations have been divided by 2 before encoding , whereas in others the original elevations are used. In general, the entropy is smaller if the elevations have been divided by 2.

Tables 1A and 1B illustrate the effect of applying the Huffman encoding algorithm

to ST06 and ST08 after using the triangular prediction algorithm [2] when banding is used. A further improvement of up to 7% is possible by the use of the Lagrange Multiplier method [3]. It is evident that the blocking method is successful in reducing the average code length when the code efficiency is otherwise low.

2.2. Comparative Results using the Two Dimensional Discrete Cosine Transform Algorithm with Blocking for Huffman Encoding.

Discrete Cosine Transform techniques are data independent and samples in the transform domain are selected, quantised and coded according to the number of bits needed for compression. Here acceptable results are obtained for a wide range of compression ratios. Several studies were made using the Discrete Cosine Transform. Various subgrids were selected from ST06 and ST08 typically of size 16,32 and 64 square and the 2D-DCT and its inverse applied to increasingly varied terrain topography. The reconstructed terrain was compared graphically with the original. Further compression can be achieved by Huffman encoding the reduced set of coefficients. The errors introduced on decompression by the application of the inverse transform were variable with the largest range in height error appearing with the greatest compression ratio. Moreover, there was no visible detectable structure or linear relationship in the reconstructed errors but the source of the greatest error was associated with sharp changes in terrain profile i.e. valley sides and coastal cliff areas.

The 2D-DCT was encoded into an Ada program written specially for this study. The results presented here are for the case when 8x8 cells are used, but other sizes can be used. Quantisation values are not used, and relative coding of DC coefficients is optional. The algorithm is also combined with Huffman encoding. The effect of blocking the DCT coefficients prior to Huffman encoding was investigated. Again the data sets used were ST06 and ST08 and the DCT operation typically produces accuracy errors when the coefficients are converted to integers before Huffman encoding and back to floating point numbers before the inverse DCT transform is applied. The Huffman encoding of coefficients itself is error-free and reversible. These accuracy errors are duly noted as is the effect of banding the coefficients before Huffman encoding into sets of size 1-4. All the results in Table 2 are from applying these two algorithms to the original terrain height data using a window of size 8x8 'pixels'. In Table 3, the methods were applied to terrain data values firstly divided by 2 to reduce the range of coefficient values produced by the DCT transform for further comparative analysis with Tables 1A and 1B. Table 4 again shows equivalent results only this time 'differencing' was applied to the DCT term in each 8x8 window along each row-block and relative to the (0,0) position. Interestingly this worked well for ST08 but not for ST06. This was due to the overall reduction in scale for the DC coefficient values for ST08 that didn't affect ST06 since ST06 already had many more zero valued coefficients. This was based on the large number of zero values for sea areas in the original terrain.

2.3. Discussion on Efficiency Comparison.

In general combining these algorithms with blocking (Tables 2 &3) produced the best results. Since we are looking at code efficiencies close to 1, blocking the symbols overcomes the decrease in code efficiency when the entropy is less than 1 bit per symbol. In ST06, the algorithms (without 'differencing') produced the lowest entropy and average code length of 0.4540 and 0.5620 of a bit per coefficient (and hence bit per elevation) respectively. This is equivalent to a storage saving of 96.49% with a decompression error due to storage accuracy of the DCT coefficients of ±5 metres, a mean of 0.0011, an rmse of 0.5210 and a standard deviation of 0.5210 (with MINITAB™). Furthermore, one can compare with the prediction plus Huffman method of Tables 1A & 1B which have error values of ±6 metres, a mean of 1.0178, an rmse of 2.4042 and a standard deviation of 2.1715. This corresponds to an entropy value of 0.5013, an average code length of 0.5507 bits per elevation and a storage saving of 96.56%. The error distribution for the former DCT hybrid methods tend to be gaussian in form with the peak becoming more rounded as the coefficients are grouped into larger sets. In the latter prediction hybrid methods to include errors, the error profile can be said to be 'flat' in form. In ST08, the best performance for comparative measures comes from Table 4. Using the DCT method and a blocksize of 4 for Huffman encoding of the coefficients, an entropy of 0.9091 and an average code length of 0.9409 bit per coefficient (bit per elevation) corresponds to a storage saving of 94.12%. Noted is the fact that the efficiency of the Huffman encoding algorithm decreases when the block size is greater than 3 for ST08. The error profile gives a maximum error range of about 5 metres, a mean of 0.003, an rmse of 0.7898 and a standard deviation of 0.7893 (see below). Here, for comparative results we must look at Table 4B and an error range of ± 4 (actual 8) metres. In this case, blocking by 4 to give an entropy of 0.9383 bit per elevation and an average code length of 0.9450 bits per elevation achieves a storage saving of 94.09%. However, the overall errors are much bigger: the rmse is 2.5974 and the standard deviation is 2.5909.

The conclusion to be drawn from these results is that for lossy compression, the DCT/Huffman method gives the best compromise between compression and error, but only when blocking takes place before Huffman encoding. This may however, create a computational overhead that makes the method unattractive in practice.

Huffman Encoding with Allowable Errors, using Error Banding and the Triangular Prediction Algorithm [3].

(Heights/ 2)

ST06

Error Range ± m (Rmse)	Blocksize	Entropy (bits /elevation)	Average Code Length (bits /elevation)	Code Efficiency (%)	Storage Saving (%)
1	1	1.3910	1.5642	88.9304	90.2239
	2	1.3273	1.3443	98.7376	91.5984
(0)	3	1.2760	1.2807	99.6340	91.9957
	4	1.2268	1.2313	99.6328	92.3041
2	1	0.8914	1.2789	69.7018	92.0069
	2	0.8418	0.9251	90.9994	94.2183
(0.8904)	3	0.8059	0.8334	96.6923	94.7910
	4	0.7823	0.7938	98.5473	95.0387
4	1	0.7052	1.2003	58.7541	92.4980
	2	0.6613	0.7983	82.8448	95.0108
(1.6860)	3	0.6319	0.6888	91.7354	95.6947
	4	0.6114	0.6411	95.3641	95.9931
6	1	0.5809	1.1535	50.3590	92.7905
	2	0.5434	0.7291	74.5297	95.4433
(2.4042)	3	0.5167	0.6034	85.6364	96.2287
	4	0.5013	0.5507	91.0170	96.5579
8	1	0.4936	1.1231	43.9531	92.9809
	2	0.4598	0.6840	67.2177	95.7250
(3.1391)	3	0.4361	0.5499	79.3044	96.5632
	4	0.4236	0.4918	86.1303	96.9260
10	1	0.4186	1.0991	38.0836	93.1303
	2	0.3861	0.6473	59.6486	95.9545
(3.9149)	3	0.3694	0.5072	72.8387	96.8301
	4	0.3551	0.4419	80.3544	97.2380

Table 1A.

Huffman Encoding with Allowable Errors, using Error Banding and the Triangular Prediction Algorithm [3].
(Heights / 2)

ST08

Error Range ± m (Rmse)	Blocksize	Entropy (bits /elevation)	Average Code Length (bits /elevation)	Code Efficiency (%)	Storage Saving (%)
1	1	2.3689	2.4386	97.1401	84.7585
	2	2.3376	2.3563	99.2034	85.2730
(0)	3	2.3043	2.3133	99.6112	85.5416
	4	2.2444	2.2513	99.6947	85.9293
2	1	1.5201	1.6559	91.8011	89.6509
	2	1.4780	1.5084	97.9854	90.5724
(0.8173)	3	1.4531	1.4621	99.3861	90.8619
	4	1.4342	1.4435	99.3570	90.9784
4	1	1.3074	1.5187	86.0840	90.5082
	2	1.2456	1.2700	98.0804	92.0626
(1.4549)	3	1.2139	1.2364	98.1864	92.2728
	4	1.1900	1.1963	99.4770	92.5234
6	1	1.1697	1.4334	81.6063	91.0412
	2	1.1012	1.1316	97.3092	92.9273
(2.0005)	3	1.0664	1.0758	99.1217	93.2762
	4	1.0394	1.0548	98.5455	93.4077
8	1	1.0667	1.3738	77.6409	91.4136
	2	1.0015	1.0440	95.9312	93.4751
(2.5974)	3	0.9649	0.9709	99.3886	93.9322
	4	0.9383	0.9450	99.2922	94.0937
10	1	0.9742	1.3247	73.5369	91.7204
	2	0.9121	0.9726	93.7827	93.9214
(3.9149)	3	0.8751	0.8841	98.9866	94.4744
	4	0.8500	0.8537	99.5640	94.6643

Table 1B.

Two Dimensional Discrete Cosine Transform (‡)with Huffman Encoding of Coefficients.
(Original Heights, 8x8 windows)

ST06

Block-size	Entropy (bits/ Coefficient)	Average Code Length (bits/ Coefficient)	Code Efficiency (%)	Storage Saving (%)
1	0.998178	1.4731	67.7598	90.7930
2	0.840046	1.0229	82.1276	93.6072
3	0.835856	0.9178	91.0756	94.2640
4	0.620969	0.6870	90.3936	95.7065

Error Range (metres) on Reconstruction through Rounding before Huffman Encoding:-

Max = 5.0000
Min = -5.1875
Mean = - 0.003
Rmse = 0.6716

ST08

Block-size	Entropy (bits/ Coefficient)	Average Code Length (bits/ Coefficient)	Code Efficiency (%)	Storage Saving (%)
1	2.069000	2.2048	93.8423	86.2200
2	1.775474	1.7967	98.8210	88.7709
3	1.742708	1.7478	99.7080	89.0762
4	1.248854	1.2528	99.6853	92.1700

Error Range (metres) on Reconstruction through Rounding before Huffman Encoding:-

Max = 4.7082
Min = -4.4846
Mean = - 0.0013
Rmse = 0.9726

Table 2.

Two Dimensional Discrete Cosine Transform (‡) with Huffman Encoding of Coefficients
(Heights/2, 8x8 Windows)

ST06

Block-size	Entropy (bits/ Coefficient)	Average Code Length (bits/ Coefficient)	Code Efficiency (%)	Storage Saving (%)
1	0.650300	1.2818	50.7309	91.9885
2	0.541816	0.8142	66.5469	94.9113
3	0.554087	0.6928	79.9790	95.6701
4	0.454014	0.5620	80.7887	96.4876
Error Range (metres) on Reconstruction through Rounding before Huffman Encoding:-			Max = 5.0939 Min = -5.0300 Mean = 0.0011 Rmse = 0.5210 148630 Zero Coefficients	

ST08

Block-size	Entropy (bits/ Coefficient)	Average Code Length (bits/ Coefficient)	Code Efficiency (%)	Storage Saving (%)
1	1.401504	1.7376	80.6574	89.1400
2	1.178912	1.2883	91.5101	91.9482
3	1.180569	1.2142	97.2263	92.4109
4	0.909866	0.9417	96.6206	94.1144
Error Range (metres) on Reconstruction through Rounding before Huffman Encoding:-			Max = 6.1186 Min = -4.5141 Mean = 0.0030 Rmse = 0.7898 132146 Zero Coefficients	
‡Typical Calculation Time for Slow Algorithm: Forward DCT 5m 1sec. , Inverse DCT 5m 46 secs.				

Table 3.

Two Dimensional Discrete Cosine Transform (‡) with Huffman Encoding of Coefficients .

(Heights/ 2, 8x8 Windows with DC Term Row Differencing)

ST06

Block-size	Entropy (bits/ Coefficient)	Average Code Length (bits/ Coefficient)	Code Efficiency (%)	Storage Saving (%)
1	0.6591	1.2874	51.1960	91.9539
2	0.5531	0.8226	67.2388	94.8587
3	0.5652	0.7017	80.5411	95.6142
4	0.4650	0.5708	81.4715	96.4327

Error Range (metres) on Reconstruction through Rounding before Huffman Encoding:-	Max = 5.0939 Min = -5.0300 Mean = 0.0011 Rmse = 0.5210 148496 Zero Coefficients

ST08

Block-size	Entropy (bits/ Coefficient)	Average Code Length (bits/ Coefficient)	Code Efficiency (%)	Storage Saving (%)
1	1.3941	1.7303	80.5700	89.1856
2	1.1755	1.2848	91.4876	91.9697
3	1.1777	1.2114	97.2189	92.4290
4	0.9091	0.9409	96.6204	94.1194

Error Range (metres) on Reconstruction through Rounding before Huffman Encoding:-	Max = 6.1186 Min = -4.5141 Mean = 0.0030 Rmse = 0.7898 132146 Zero Coefficients

‡Typical Calculation Time for Slow Algorithm:
Forward DCT 5m 1sec. , Inverse DCT 5m 46 secs.

Table 4.

3.0. The Use of Wavelet Transforms for the Data Compression of DEM's.

The key idea with wavelet transforms is in the formation of classes of signals into weighted sums of basis functions (complex exponentials for the Fourier Transform and cosines for the Cosine Transform). In contrast to traditional Fourier theory, the basis functions are formed by scaling and translating a single function and the mathematical properties of the decomposition are determined by the properties of the underlying function. Thus unlike sines and cosines, which define an unique Fourier or Cosine transform, there is not one single unique set of wavelets; in fact, there are infinitely many possible sets. A particular set of wavelets is specified by a particular set of numbers, called wavelet filter coefficients. One such simple set comes from a class discovered by Daubechies [4] which include members ranging from being highly localised to highly smooth. Press [5] describes both the transformation methods for the simple case and how the Discrete Wavelet Transform (DWT) is formalised. Compact (and therefore unsmooth) wavelets are better for lower accuracy approximation and for functions with discontinuities (like edges), while smooth (and therefore non–compact) wavelets are better for achieving high numerical accuracy. By taking a multi–dimensional wavelet transform of an image, compression is achieved by bit allocation amongst the coefficients in some highly non–uniform, optimised way. In general, large wavelet coefficients are quantised accurately, whilst small coefficients are quantised coarsely with only a bit or two or may even be truncated completely. If the resulting quantisation levels are still statistically non–uniform, they may then be further compressed by a technique such as Huffman encoding. When a smooth and coarser wavelet transform are applied to a DEM, the performance is much the same in terms of reconstruction error from accuracy and compression ratio. There is however a slight improvement in computation time.

Two Dimensional 4-Coefficient ('Localised') Daubechies Wavelet Transform with Huffman Encoding of Coefficients.

(Original Heights)

ST08

256x256 subset

Block-size	Entropy (bits/ Coefficient)	Average Code Length (bits/ Coefficient)	Code Efficiency (%)	Storage Saving (%)
1	3.3885	3.4170	99.1654	78.6438
2	3.0484	3.0666	99.4068	80.8335
3	2.6881	2.6994	99.5842	83.1290
4 •	1.8727	1.8807	99.5721	88.2456

Invertible Wavelet Transform.
Error Range (metres) on Reconstruction through **Rounding** before Huffman Encoding:-

Max = 3.0
Min = -2.0
Mean = 0.50
Rmse = 0.79
Stdev = 0.61

32093 'Zero' Coefficients, ST08 Coordinates (10,10).
Typical Transform Calculation time 20 secs.
• Typical computation time 3 mins (DEC Alpha).

Table 5.

Two Dimensional 12-Coefficient ('Smooth') Daubechies Wavelet Transform with Huffman Encoding of Coefficients.

(Original Heights/2)

ST06

256x256 subset

Block-size	Entropy (bits/ Coefficient)	Average Code Length (bits/ Coefficient)	Code Efficiency (%)	Storage Saving (%)
1	1.0261	1.5070	68.0902	90.5815
2	0.8619	1.0489	82.1721	93.4443
3	0.7425	0.8438	88.0017	94.7265
4 •	0.5383	0.6335	84.9665	96.0406

Invertible Wavelet Transform.
Error Range (metres) on Reconstruction through **Rounding** before Huffman Encoding:-

Max = 2.0
Min = -3.0
Mean = -0.26658
Rmse = 0.56420
Stdev = 0.49784

57829 'Zero' Coefficients, ST06 Coordinates (0,0).
Typical Transform Calculation time 20 secs.

• Typical computation time 3 mins (DEC Alpha)

Table 6.

Two Dimensional 12-Coefficient ('Smooth') Daubechies Wavelet Transform with Huffman Encoding of Coefficients.
(Original Heights/2)

ST06
256x256 subsets

Block-size	Entropy (Bits/ Coefficient)	Average Code Length (Bits/ Coefficient)	Code Efficiency (%)	Storage Saving (%)
Coordinates (0,0)				
1	1.0261	1.5070	68.0902	90.5815
2	0.8619	1.0489	82.1721	93.4443
3	0.7425	0.8438	88.0017	94.7265
4 •	0.5383	0.6335	84.9665	96.0406
Coordinates (0,144)				
1	0.9633	1.4743	65.3437	90.7859
2	0.8001	1.0047	79.6315	93.7203
3	0.6830	0.7983	85.5519	95.0103
4	0.4817	0.5833	82.5810	96.3542
Coordinates (144,0)				
1	1.3991	1.7360	80.5957	89.1500
2	1.1993	1.3067	91.7783	91.8329
3	1.0436	1.0935	95.4397	93.1658
4	0.7606	0.8132	93.5356	94.9174
Coordinates (144,144)				
1	1.3251	1.6937	78.2379	89.4145
2	1.1294	1.2551	89.9883	92.1556
3	0.9791	1.0402	94.1217	93.4985
4	0.7308	0.7924	92.2235	95.0475
Typical Transform Calculation time 20 secs.			• Typical Computation time 3mins. (DEC Alpha)	
Invertible Wavelet Transform. Error Range (metres) on Reconstruction through **Rounding** before Huffman Encoding:-			Max = 2.0 Min = -3.0 Mean = -0.26658 Rmse = 0.56420 Stdev = 0.49784	

Table 7.

Two Dimensional 4-Coefficient ('Localised') Daubechies Wavelet Transform with Huffman Encoding of Coefficients.
(Original Heights/2)

ST08
256x256 subsets

Block-size	Entropy (Bits/ Coefficient)	Average Code Length (Bits/ Coefficient)	Code Efficiency (%)	Storage Saving (%)
Coordinates (0,0)				
1	2.2938	2.4014	95.5188	84.9915
2	2.0289	2.0447	99.2277	87.2208
3	1.7923	1.8012	99.5079	88.7426
4 •	1.2202	1.2268	99.4639	92.3325
Coordinates (0,144)				
1	2.2078	2.3358	94.5171	85.4010
2	1.9512	1.9723	98.9299	87.6732
3	1.7220	1.7307	99.4990	89.1834
4 •	1.2285	1.2377	99.2570	92.2645
Coordinates (144,0)				
1	2.5926	2.6566	97.5900	83.3962
2	2.3031	2.3140	99.5265	85.5372
3	2.0367	2.0517	99.2679	87.1770
4 •	1.3939	1.3970	99.7746	91.2684
Coordinates (144,144)				
1	2.2902	2.4022	95.3366	84.9862
2	2.0291	2.0459	99.1780	87.2130
3	1.7932	1.8020	99.5095	88.7375
4 •	1.2568	1.2636	99.4646	92.1026

Typical Transform Calculation time 20 secs.	• Typical Computation time =3 mins (DEC Alpha)
Invertible Wavelet Transform. Error Range (metres) on Reconstruction through **Rounding** before Huffman Encoding:-	Max = 3.0 Min = -2.0 Mean = 0.50433 Rmse = 0.7879 Stdev = 0.60534

Table 8.

3.1. Results.

Both sets of test data were partitioned into subsets of 256x256 arrays and both the two-dimensional 'localised' 4-coefficient and the 'smooth' 12-coefficient Daubechies wavelet transform were applied in turn to each data set. In all cases the resulting coefficient matrices were Huffman encoded with statistical analyses made as described above. The only errors on reconstruction were due to the rounding of the coefficients before Huffman encoding leading to some loss of accuracy on reconstructing the original data.

Table 5 shows the results of applying the 'localised' Daubechies wavelet transform to a typical subset of ST08 using the original terrain heights in the data vector. This 'localised' transform performed better than the 'smoother' transforms (12 and 20 coefficient) on the rougher terrain data and the results for storage savings can therefore be compared to using the Two-Dimensional Discrete Cosine Transform (2D-DCT) in Table 2. This method together with blocking the coefficients before Huffman encoding achieves an average code length of 1.8807 bits per coefficient compared with 1.2528 bits per coefficient for the 2D-DCT case. This in turn gives the comparative storage savings of 88.25% and 92.17 respectively. The advantage that the wavelet method has over the DCT method is in the efficiency of the algorithm, it is much faster even when blocking is done. In the same light, Table 6 where the original heights of ST06 are divided by 2, can be compared to the results in the top half of Table 3. For ST06, the 'smoother' 12-coefficient wavelet transform worked best. Although, the matrices are of differing sizes, again the results are marginally inferior in terms of average code length but the final storage savings are about the same (96%). Tables 7 and 8 describe in full the results for the four 256x256 overlapping squares from ST06 and ST08 for original terrain data divided by 2, the smother wavelet working better on the smoother terrain of ST06 and the more localised wavelet best on the more varied landscape of ST08.

The results overall are marginally worse than with the 2D-DCT if one compares Tables 2 & 3 with Tables 6 or Table 7. Indeed when the accuracy errors are compared with Tables 4A and 4B where errors are introduced (through banding) into the data prior to the prediction algorithm and Huffman encoding, still gives the prediction method an advantage over the transformation methods. For example, in a typical subset of ST08, the 4-coefficient Daubechies wavelet transform with Huffman encoding of the coefficients (when the coefficients are blocked into sets of four), gives an entropy value of 1.8727 bits per coefficient and a storage saving of 88.25%. This is when the min./max. error is 3 and -2 metres and the rmse is 0.79 (Table 12). Table 4B shows the equivalent performance when the prediction algorithm and error banding is used for ST08. In this case for a max./min. error of 2 metres and an rmse of 0.8173, comparative results with blocking sets of prediction errors into sets of 4 gives an entropy of 1.4342 bits/elevation and a storage saving of 90.98%. Although the rmse values are about the same, the range of errors is much smaller.

4.0. Summary of Research.

Comparisons have been made between these predictive encoding methods and a transform coding technique, the Two-Dimensional Discrete Cosine Transform (2D-DCT). Transform coding allows greater compression but is computationally intensive and is subject to a greater degree of error on reconstruction of the data. The Discrete Cosine Transform coding method can be combined with either Huffman encoding or Run Length Encoding (RLE) of the DCT coefficients to achieve greater compression. Further compression can be achieved when using the Huffman DCT method by blocking the transformation coefficients. The method of blocking the DCT coefficients before Huffman encoding gives a better performance than Run Length encoding of the DCT coefficients The best compression achievable for one data set (ST06) using the DCT algorithm with blocking is about 35.24:1, i.e. a storage saving of 96.49% for at most an error of 5 metres and a root mean square error (rmse) of 0.5. For error-free compression (accurate to the nearest metre), the simple prediction method [2] gives a compression ratio of 13.04:1 with blocking the prediction errors before Huffman encoding. This gives a storage saving of about 92.30%. Similar results for a second more variable data set (ST08) using the DCT algorithm, give a compression ratio of 17.60:1 or a storage saving of 94.12%, again for an equivalent maximum error of about 5 metres and an rmse of about 0.8. In the error-free Huffman method with blocking, equivalent results for the second data set (ST08) are a compression ratio of about 7.13:1 and a storage saving of 85.93%. The wavelet transform method tested produces similar but marginally inferior results compared to the 2D-DCT. The Lagrange Multiplier method [3] will give an improvement of about 7% to the error-free compression ratios quoted. There remain a large number of possible variations on the method, including the use of arithmetic, adaptive Huffman or adaptive arithmetic coding in place of static Huffman encoding. Wavelet transforms give similar results to the DCT but are much more efficient. Since both these algorithms are computationally expensive, a trade-off between maximum compression ratio and speed of compression/decompression must be made.

Current work involves looking at different families of wavelets for compression and developing efficient heuristic algorithms to selectively constrain the terrain topography to improve prediction methods.

[1] Huffman D.A. "A Method for the Construction Of Minimum Redundancy Codes". Proc. IRE, Vol. 40, 1952. pp.1098-1101.

[2] Kidner D.B. & Smith D.H. "Compression of Digital Elevation Models by Huffman Encoding"; Computers & Geosciences, Vol. 18., No. 8, pp.1013-1034, 1992.

[3] Lewis M. & Smith D.H. "Optimal Predictors for the Data Compression of Digital

Elevation Models Using the Method of Lagrange Multipliers". pp. 246-256. Proc Auto Carto 11, Oct. 30-Nov 3rd 1993, Minneapolis, Minnesota,USA.

[4] Daubechies I., 1988 Commun. Pure & Applied Maths, vol.44, pp.141–183.

[5] Press W.H., "Wavelet Transforms", 2nd Edition of Numerical Recipes:The Art of Scientific Computing. Harvard–Smithsonian Center for Astrophysics. Cambridge Mass.

[6] Nelson M.R. The Data Compression Book. Prentice Hall 1991.

Radiometric Resolution Enhancement by Lossy Compression as compared to Truncation Followed by Lossless Compression

J. C. Tilton
Code 935
Goddard Space Flight Center
Greenbelt, MD 20771

M. Manohar
Hughes STX, Code 935
Goddard Space Flight Center
Greenbelt, MD 20771

Abstract

Recent advances in imaging technology make it possible to obtain imagery data of the Earth at high spatial, spectral and radiometric resolutions from Earth orbiting satellites. The rate at which the data is collected from these satellites can far exceed the channel capacity of the data downlink. Reducing the data rate to within the channel capacity can often require painful trade-offs in which certain scientific returns are sacrificed for the sake of others. In this paper we model the radiometric version of this form of lossy compression by dropping a specified number of least significant bits from each data pixel and compressing the remaining bits using an appropriate lossless compression technigue. We call this approach "truncation followed by lossless compression" or TLLC. We compare the TLLC approach with applying a lossy compression technique to the data for reducing the data rate to the channel capacity, and demonstrate that each of three different lossy compression techniques (JPEG/DCT, VQ and and Model-Based VQ) give a better effective radiometric resolution than TLLC for a given channel rate.

1 Introduction

The imaging sensors onboard satellites are capable of scanning the Earth at very high spatial, spectral and radiometric resolutions. Downlink channel capacity is often a major limiting factor for the resolution at which the data is collected. Image compression techniques can be used to reduce the data rate from the imaging sensor to within the downlink channel capacity.

Ideally, decompression of the downlinked data should result in the full lossless recovery of the image data as sensed onboard the satellite. However, the amount of compression possible from lossless techniques is bounded by the entropy of the source. This entropy bound limits the amount of compression that can be obtained to the range of 2 to 3 for most NASA image data sources. This is most often insufficient to reduce the sensor data rate to within the channel capacity.

Large amounts of compression can, instead, be obtained with lossy compression techniques. In fact, a crude form of lossy compression is most often used in these cases, i.e. the temporal, spatial, spectral, and/or radiometric resolutions are limited to produce a data rate that can be handled by the channel capacity. Establishing these limits often requires painful trade-offs in which certain scientific returns are sacrificed for the sake of others. In this paper we model the radiometric version of this form of lossy compression by truncating a specified number of least significant bits followed by lossless compression of the remaining higher order bits. We call this approach "Truncation followed by Lossless Compression" (TLLC). Using the TLLC approach, the data rate can be set to within the channel capacity by selecting the appropriate number of least significant bits dropped. We have found that this method produces reasonable rate distortion values for compression ratios less than 5 or 6. However, for larger compression ratios, the rate distortions increase exponentially as the amount of truncation increases.

Much better rate distortion behavior can be obtained by using other lossy compression approaches. For the lossy compression approaches we have studied, the rate distortion performance is either linear or sublinear. These lossy compression approaches are the JPEG/DCT (Joint Photographic Experts Group/Discrete Cosine Transform [1]), VQ (Vector Quantization [2], and the more recently developed MVQ (Model-based VQ [3]) approach. For a given data rate, this improved distortion behavior over TLLC can be looked upon as a gain in radiometric resolution.

We first describe the TLLC approach in more detail, and give summary descriptions of the JPEG/DCT, VQ and MVQ lossy compression approaches. We then derive our measure of gain in radiometric resolution of a particular lossy compression approach over TLLC. Finally we demonstrate the gain in radiometric resolution provided by the JPEG/DCT, VQ and MVQ appproaches over the TLLC approach with imagery data from three remote sensing instruments: the Landsat Thematic Mapper (TM), the Advanced Solid-state Array Spectroradiometer (ASAS), and the Advanced Very High Resolution Radiometer (AVHRR). Of these, TM imagery data is at 8-bit resolution, while imagery data from the other two are at 12-bit pixel resolution with at most 10 significant bits.

2 Lossy Image Compression Techniques

Lossy compression can produce relatively high compression ratios or low data rates (bit rates) at a cost of losing some information. Here we define the compression ratio (CR) to be the ratio of the number of bits in the original image to the number of bits in the compressed image. The bit rate in bits/pixel can be represented as n/CR, where n is the radiometric resolution (in bits/pixel) of the original image. A common measure of information loss or distortion is the mean squared error between the original image and the image reconstructed from the compressed data. The mean squared error is defined formally as

$$MSE = \frac{1}{N} \sum_{k=0}^{N-1} (f_1(k) - f_2(k))^2 \quad (1)$$

where $f_1(k)$ and $f_2(k)$ are the k^{th} pixels from the original and reconstructed images, respectively, and N is number of pixels in the image. The performance of a lossy compression technique can be characterized by a rate-distortion curve, which is simply a plot of bit rate (n/CR) versus distortion (MSE).

In the following subsections we describe the TLLC approach and other lossy compression techniques that we have used in our tests.

2.1 Truncation followed by Lossless Compression (TLLC)

Truncation followed by Lossless Compression (TLLC) is not a compression approach that one would use directly. However, as mentioned in the introduction, it is a model for the design practice of setting the radiometric resolution to a lower value than sensor technology would allow, so as to keep the data rate produced by the sensor within the limits of channel capacity for bringing the data from the sensor to Earth.

Let the radiometric resolution of the image data collected at the instrument be n bits/pixel and the channel capacity be m bits/pixel ($m < n$). The TLLC approach reduces the bit rate from n to no more than m by dropping a number of lower order bits b. Here b is chosen such that the lossless compression of remaining n-b bits results in an output bit rate of no more than m bits/pixel. The lossless compression approach that consistently performed best in the cases we tested utilizes the coding model for lossless encoding specified in the JPEG still image compression standard [1] combined with the Witten-Neal-Cleary version of arithmetic coding [9].

2.2 JPEG/DCT

JPEG/DCT([1]) lossy compression algorithm consists of three successive stages: Discrete Cosine Transform (DCT) transformation, coefficient quantization and lossless compression. The original image is partitioned into nonoverlapping 8x8 pixel blocks. Each block is independently transformed using the DCT. The DCT coefficients are then quantized using a quantization table that is designed using the Human Visual System (HVS) contrast sensivity function. The first coeffient of DCT transformation is DC coefficient and is proportional to average brightness of the block. The quantized DC coefficient along with other DC coefficients is compressed using DPCM (Differential Pulse Code Modulation) using 1-D causal prediction. The quantized AC coefficients are zig-zag scanned to covert 2-D array into 1-D array and then are lossless compressed by using Huffman table that is transmitted to the decoder as a part of the header information.

The baseline JPEG/DCT does not include standards for pixel resolutions higher than 8-bits. Since some of the images tested here have 12 bit resolution, we truncated the image pixels such that the pixel resolution after truncation was 8-bits. After JPEG/DCT compression was applied and the image was reconstructed from the compressed data, each pixel value was multiplied by the truncation scale factor to scale the pixels values properly for MSE measurements.

Spectral correlations are not easy to exploit in JPEG/DCT, as there are no standards for decorrelating the bands of multispectral image data (JPEG/DCT does however, allow red, green and blue decorrelations by converting them to luminance and chrominance components. ([1], pp.18-20, p.503). Therefore, we compressed each band of the multispectral images independently in our tests.

2.3 Vector Quantization

Vector Quantization (VQ) is the vector extension of scalar quantization which is found to be very useful for multispectral image compression ([4] [5]). The VQ vectors are obtained from image data by systematically extracting nonoverlapping blocks (typically 4x4) and arranging the pixels in each block in raster scan order. Such vectors allow VQ to exploit two dimenstional correlations in the image data. If the image is multispectral, nonoverlapping cubes (typically 4x2x3) may be used. VQ builds up a dictionary of a few representative vectors, called codevectors, and then codes the image with the index value of the closest codevector from the dictionary, called the codebook, in place of of each vector. Each codevector is represented by an address containing $log_2 M$ bits, where M is number of codevectors in the codebook. Assume vectors of size k are drawn from the input image and matched with those in the codebook. Using the indices of the matched codevectors to represent the input image vectors results in a decreased rate of $(log_2 M)/k$ bits/pixel or a compression ration of

$(k * n)/log_2 M$, where n is the radiometric resolution of the image. In all practical situations the codebook size, M, is much smaller than the number of vectors that make up the input image.

The most important phase of VQ is the training process in which an optimal codebook (by some criterion such as least MSE) is learned from the input samples. The most widely used algorithm is Linde-Buzo-Gray (LBG) algorithm ([6]). Both the training and coding phases of VQ require finding the codevector which is closest match to a given vector. Computing this closest match requires computations proportional to the size of the codebook. Computational cost can be reduced by employing a suboptimal approaches such as Tree Search Vector Quantization (TVSVQ) and Pruned Tree VQ (PTVQ) ([7]). The computational problems can also be solved by using a special architectures ([4]). While the codebook training and data encoding steps of VQ are computationally intensive, the decoding step is not, because it is a table lookup process that can be performed quickly on a conventional sequential computers. Obvious drawbacks of VQ are computationally intensive training process for generating codebooks for a given class of images and the maintainance of these codebooks at coding and decoding ends. At the encoding end a codebook has to be selected for the given data and a pointer to this codebook may be provided as a part of the header record in the compressed file for the decoder to use the same codebook for decoding purposes. This is one practical difficulty of using VQ for image compression. This problem is solved with the Model-based Vector Quantization (MVQ) approach, described in the next section, in which codebooks are generated using statistical models and input image covariance matrix.

2.4 MVQ

In the MVQ, the codebook is generated using a statistical model of mean removed residual of the vectors. The mean removed vector elements are characterized either Gaussian or Laplacian error models. For small vectors sizes of 2 or 4, the mean removed vector elements can be simulated by a uniform random number generator producing independent and identically distributed (i.i.d) random numbers and then passing them through a Laplacian filter with mean λ. This is a reasonable model of generating mean removed residuals for these small vector sizes . However, as the vector size increases, the mean removed vector elements cannot be treated as independent and so a covariance structure of the source is imposed on Laplacian i.i.d process. For k-element vectors, the covariance matrix, Σ, of the input image is a kxk matrix. The diagonal elements of Σ are approximately equal and correspond to the variance of the normalized pixel values in the image. The square root of Σ_{00} $(= \lambda)$ is used to generate independent and identically distributed (i.i.d) Laplacian random variables. The consecutive Laplacian i.i.d random numbers are grouped into vectors of size $k = (k1 \text{ x } k2)$ to form a vector W_i (i^{th} vector). The covariance matrix, Σ, of the source is then factorized into L and U, where L and U are upper and lower triangular matrices, respectively. The factorization is performed by using the Cholesky decomposition algorithm. When the Laplacian vectors are mapped onto L, the resulting vectors will have same multivariate distribution

as Σ. The vectors thus generated are independent of other vectors. However, the vector elements have the correlations given by Σ. Let W_i be the k-element vector generated by Laplacian i.i.d process. Let the L be the lower triangular matrix obtained by Cholesky's decomposition of Σ. Now the codevector X_i (which is i^{th} codebook entry) is given by

$$X_i = L * W_i$$

These vectors are used as the code vectors for the source mean removed residual vectors. In the second pass input image is coded using the model codebook. The codebook is completely specified by a seed point of uniform random number generator, λ, and the lower triangular matrix, L. The lower triangular matrix will have at most (k2 + k)/2 nonzero real numbers, where k is the size of the vector. Thus, by transmitting seed point of the uniform random number generator, λ, and L in the header of coded file, the decoder can generate the codebook to decode the VQ coded image.

3 Radiometric Resolution Gain of Lossy Compression Algorithms

In the TLLC approach, the radiometric resolution of the input image is explicitly reduced by b bits by the truncation process. We show here that the MSE distortion resulting from the truncation varies exponentially with b, the loss in radiometric resolution. The relation between MSE distortion and loss of radiometric resolution can be derived as follows:

When b lower order bits are dropped, the error in pixel may be one of the integers (0, 1, 2. ..., 2^b-1). Assuming a uniform distribution of these error pixel values, the expected mean squared error (MSE) is given by

$$MSE = \frac{1}{2^b - 1} \sum_{k=1}^{2^b-1} k^2 \tag{2}$$

$$= (2 * 2^{2b} - 3 * 2^b + 1)/6 \tag{3}$$

The uniform distribution assumption holds best for lower values of b. Equation (3) can be derived from (2) using the Euler-Maclaurin summation formula [8]. From Equation (3), we can obtain b in terms of MSE by solving the quadratic equation in 2^b and taking log_2 giving:

$$b = log_2([3 + \sqrt{(48 * MSE + 1)}]/4) \tag{4}$$

Equation (4) can be used to compute the loss of radiometric resolution due to the mean squared error distortion for a give compression ratio. We can thus compare performance of lossy compression techniques in terms of radiometric efficiency. For a given compression ratio, let the MSE distortions from two lossy methods (for example, VQ and TLLC) be D_1 and D_2 , respectively. Let b1 and b2 be loss of radiometric resolutions from these methods that can be computed from Equation (4). Now if $b1 > b2$, there is gain in radiometric resolution, Δb, by using VQ instead of TLLC, which is given by

$$b1 - b2 = \Delta b = log_2 \left[\frac{3 + \sqrt{48 * D_1 + 1}}{3 + \sqrt{48 * D_2 + 1}} \right] \tag{5}$$

For large distortions Equation (5) can be simplified to give

$$\Delta b = \frac{1}{2} log_2 \left[\frac{D_1}{D_2} \right] \tag{6}$$

Using Equation (6) lossy compression techniques can be compared in terms effective radiometric gain by using one with lesser distortion than the other compression technique for a given rate. We have reported here the effective radiometric resolution gain of VQ, MVQ and JPEG/DCT with respect to TLLC.

4 Experimental Results

Three different multispectral image data sets are used in our experimentation. The first data set consists of spectral bands 1, 2, and 3 of a 2048-by-2048 pixel subimage of a Landsat Thematic Mapper (TM) scene collected in 1991 (path/row 46/28) from over the Gifford Ponchot National Forest in the state of Washington in the United States of America. The radiometric (pixel) resolution of this data is 8 bits. The second data set is the first two spectral bands from a 409x2048 pixel Global Area Coverage (GAC) data set from the Advanced Very High Resolution Radiometer (AVHRR) instrument taken from over the western pacific ocean. The pixel resolution of this data is 12 bits (stored as 16 bits per pixel). The third data set is made up of bands 22 and 23 from the Advanced Solid-state Array Spectroradiometer (ASAS) instrument. This data set also has 12 bit pixel resolution. We used for our test a 512x420 pixel image designated 92161553 from Volume 4 of the FIFE CD-ROM series ([10]).

A training data set is required for the VQ method. This training data set should be disjoint from the test data set, but should be from the same instrument with the same spectral bands and should have similar scene characteristics. We chose to use the first 512 columns of the TM data set for testing, and trained on columns 513 through 2048 (for all 2048 lines). The AVHRR data was divided into two equal parts. The first 1024 lines were used for testing, while the second 1024 lines were used for training. As mentioned above, we used bands 22 and 23 of ASAS data set 92161553 of size 512x420 for testing. For training

we used the same bands from the 512x590 pixel data set designated 92161621, the 512x600 pixel data set designate 92161631, and 512x600 data set designated 92161727. The training data was used to generate codebooks for each instrument with vector sizes of 4, 8, 16 and 32 so that compressed data at four different compression ratios could be obtained.

The JPEG/DCT compression technique used here was implemented for 8-bit pixel resolution images. To compress the 12-bit AVHRR and ASAS using JPEG/DCT, the images were first converted to 8-bit images by finding the brightest pixel (g_{max}) and scaling down all the pixels by the factor $g_{max}/255$. (MVQ can compress images of pixel resolutions 8-16, and does not need any codebooks for compression.)

The compression results on the TM data set are given in Table 1.1. The table provides MSE distortions for different compression ratios using the four different compression methods (TLLC, JPEG/DCT, VQ, and MVQ). The plots of CR vs. MSE are shown for the above four techniques on the TM data set are shown in Figure 1. The gain in radiometric resolution using JPEG/DCT, VQ and MVQ compared to TLLC are derived from the plots. For three CR's, the MSE's are measured from the plots and the Δb is computed from Equation (6). The radiometric resolution, Δb , for different CR's are given in Table 1.2 and the plots are shown in Figure 2. The results on AVHRR data are given in Table 2.1 and 2.2 and ASAS data results are given in Table 3.1 and 3.2. The rate distortion curves for AVHRR data and ASAS data using three lossy compressions compared to TLLC techniques in the plots shown in Figures 3 and 5 respectively. The gain in the radiometric resolution obtained by employing lossy compression techniques compared to TLLC are shown in Figure 4 for AVHRR data and Figure 6 for ASAS data.

Table 1.1: CR Vs. MSE on TM data

TLLC		JPEG		VQ		MVQ	
CR	MSE	CR	MSE	CR	MSE	CR	MSE
3.8	0.5	2.3	0.32	8.81	3.23	12.5	15.1
5.8	3.36	13.4	3.49	17.9	5.76	22.6	27.2
9.7	17.1	21.3	5.73	34.1	8.55	40.1	41.2
18.9	70.1	33.1	9.86	-	-	-	-
23.1	489	-	-	-	-	-	-

Table 1.2: Δb w.r.t TLLC for TM

CR	Δb w.r.t TLLC		
	JPEG	VQ	MVQ
10.0	0.95	1.20	0.46
15.0	1.48	1.60	0.65
20.0	1.65	1.75	0.76

Table 2.1: CR Vs. MSE on AVHRR data

TLLC		JPEG		VQ		MVQ	
CR	MSE	CR	MSE	CR	MSE	CR	MSE
4.0	3.5	3.5	8.5	8.4	51.6	4.6	131
5.2	17.1	17.5	237	16.4	179	8.77	443
7.2	76.1	28.2	351	37.8	400	20.0	574
10.6	323	46.3	500	-	-	-	-
17.0	1317	-	-	-	-	-	-

Table 2.2: Δb w.r.t TLLC for AVHRR

CR	Δb w.r.t TLLC		
	JPEG	VQ	MVQ
10.0	0.95	1.20	0.46
15.0	1.48	1.60	0.65
20.0	1.65	1.75	0.76

Table 3.1: CR Vs. MSE on ASAS data

TLLC		JPEG		VQ		MVQ	
CR	MSE	CR	MSE	CR	MSE	CR	MSE
5.96	0.5	12.7	6.5	8.2	10.9	7.0	22.0
8.36	3.47	22.3	16.8	15.8	23.4	12.8	81.5
12.41	17.4	35.0	28.0	33.5	37.0	30.0	100.3
19.48	77.3	53	50	-	-	40.3	200.1
32.0	304	-	-	-	-	-	-

Table 3.2: Δb w.r.t TLLC for ASAS

CR	Δb w.r.t TLLC		
	JPEG	VQ	MVQ
15.0	0.95	0.3	0.00
20.0	1.30	0.8	0.00
25.0	1.52	1.18	0.43
30.0	1.64	1.38	0.65

5 Conclusions

The data rates possible from remote sensing instruments can often far exceed the channel capacity for downlinking this data to Earth. The required data rate reduction is often obtained by reducing the resolution of the instrument. We have modeled the radiometric version of this approach by dropping a number of least significant bits and applying an appropriate lossless compression method. We refer to this technique as Truncation followed by Lossless Compression (TLLC). We have shown in our study that using lossy compression techniques such as JPEG, VQ and MVQ would give a gain in radiometric resolution compared to TLLC for a given data rate. In our experiments on Landsat TM data, we have found that radiometric resolution improvements of 1 to 1.5 bits for bit rates ranging from 0.8 - 0.5 or compression ratios of 10-20 with the VQ or JPEG techniques. Similar improvements are obtained for AVHRR data using VQ and JPEG techniques. However for ASAS data, the improvements are seen only for compression ratios exceeding 10 in the case of JPEG and VQ and 20 for MVQ.

References

[1] W. B. Pennebaker and J. L. Mitchell, "JPEG Still Image Data Compression Standard," *Van Nostrand Reinhold, NY 1993*, pp.203-206

[2] A. Gersho and R. M. Gray, "Vector Quantization and Signal Compression," *Kluwer Academic Publishers, 1991*

[3] M. Manohar and J. C. Tilton, B. Kobler and P. C. Hariharan, "Model-Based Vector Quantization," *Proc of the Data Compression Conference*, March 29-31, 1994, Snowbird, UT, p. 497.

[4] M. Manohar and J. C. Tilton, "Progressive Vector Quantization on a Massively Parallel SIMD Machine with Application to Multispectral Image Data," *IEEE Trans in Image Processing*, to appear.

[5] S. Gupta and A. Gersho, "Feature Predictive Vector Quantization of Multispectral Images," IEEE Trans on Geoscience and Remote Sensing, Vol. 30, No. 3, May 1992, pp. 491-501

[6] Y. Linde, A. Buzo, and R. M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Trans. on Comm*, COM-26, pp. 702-710, April 1978.

[7] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Optimal Pruning with Applications to Tree Structured Source Coding and Modeling," *IEEE Trans. Inform. Theory*, March 1989, pp. 299-315.

[8] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Ninth Printing, November 1970, p. 16, U. S. Goverment Printing Office, Washington, D. C.

[9] I. H. Witten, R. M. Neal and J. G. T. Cleary, "Arithmetic Coding for Data Compression," *Communications of the ACM*, Vol. 30, 1987, pp. 520-540.

[10] D. R. Landis, D. E. Strebel, J. A. Newcomer and B. W. Meeson, "Archiving the FIFE Data on CD-ROM," *Proceedings of the 1992 International Geoscience and Remote Sensing Symposium*, May 26-9, 1992, Houston, TX, pp. 65-7.

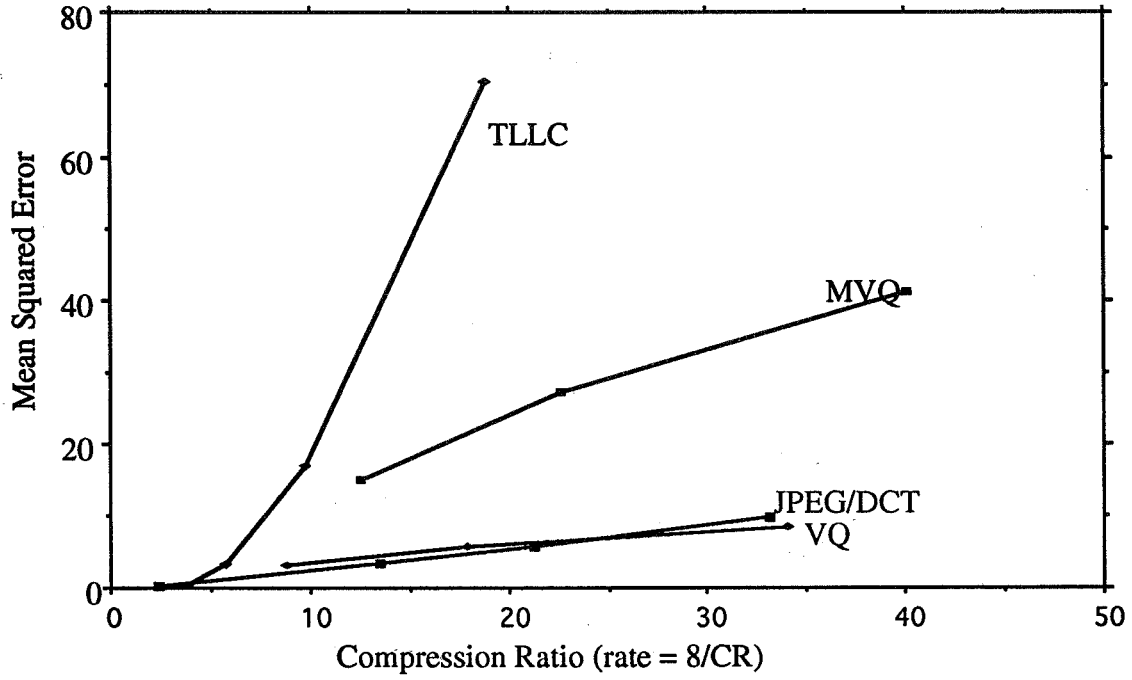


Figure 1. Rate-Distortion performance of lossy compression techniques and TLLC on the TM data set

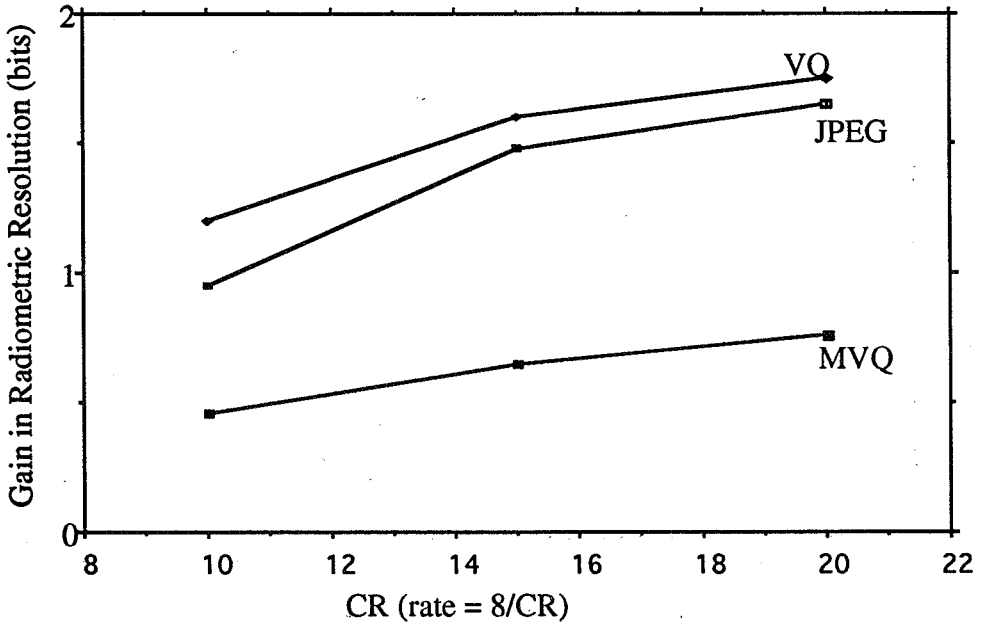


Figure 2. Radiometric Resolution of Lossy Compression techniques on the TM data set

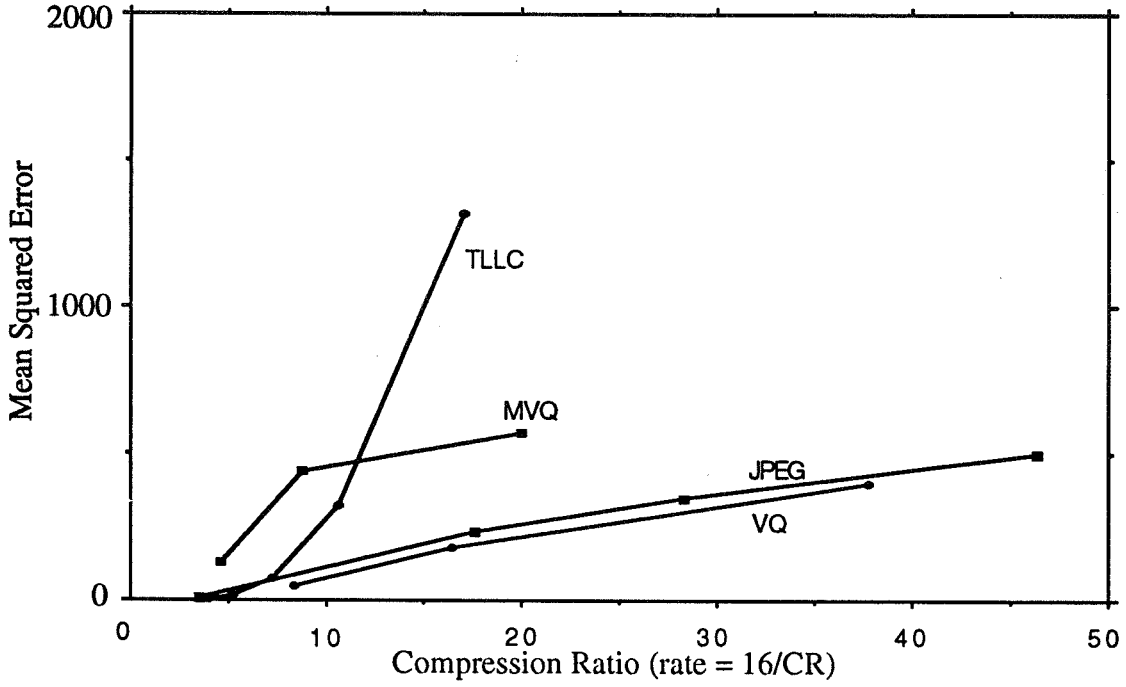


Figure 3. Rate-Distortion performance of lossy compression techniques and TLLC on AVHRR data set

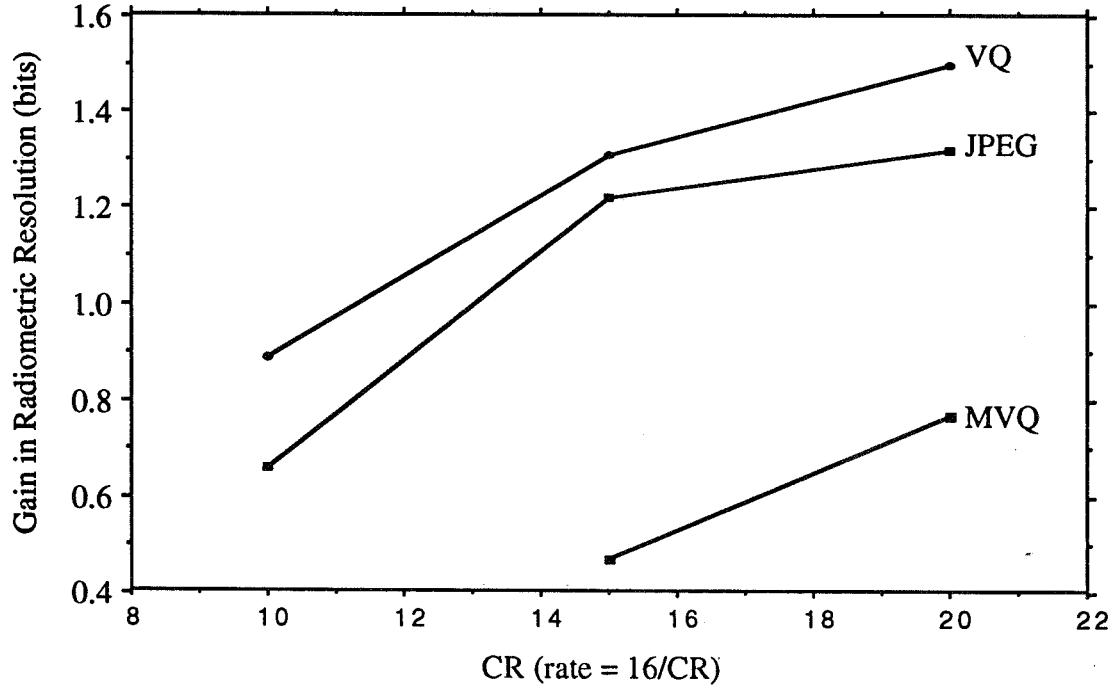


Figure 4. Radiometric Resolution Gain of Lossy Compression techniques on AVHRR data set

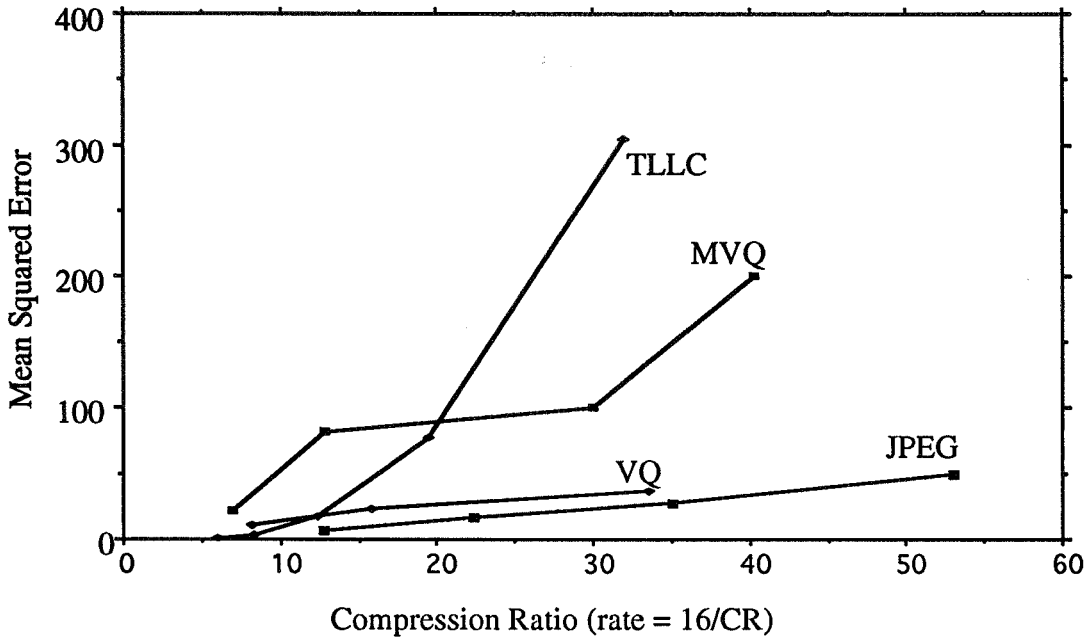


Figure 5. Rate-Distortion performance of lossy compression techniques on ASAS data set

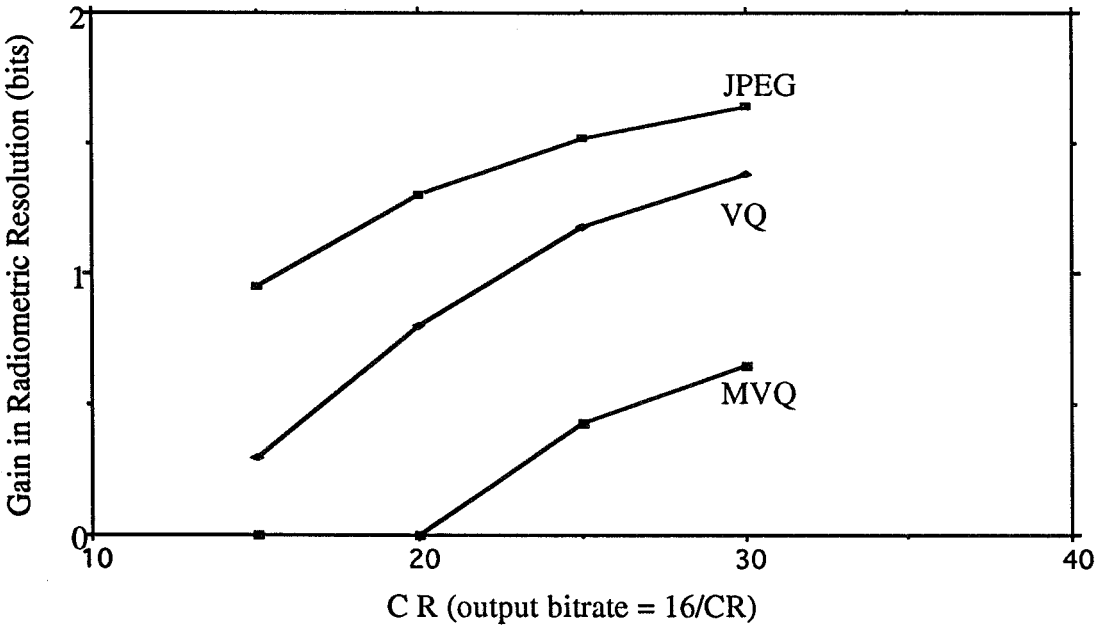


Figure 6. Radiometric Resolution of Lossy Compression techniques on ASAS data set

BLOCKING REDUCTION OF LANDSAT THEMATIC MAPPER JPEG BROWSE IMAGES USING OPTIMAL PSNR ESTIMATED SPECTRA ADAPTIVE POSTFILTERING*

Irving Linares, Russell M. Mersereau and Mark J. T. Smith
School of Electrical Engineering
Georgia Institute of Technology
Atlanta, GA 30332 USA
E-Mail: linares@defiant.gsfc.nasa.gov

ABSTRACT

Two representative sample images of Band 4 of the Landsat Thematic Mapper are compressed with the JPEG algorithm at 8:1, 16:1 and 24:1 Compression Ratios for experimental browsing purposes. We then apply the Optimal PSNR Estimated Spectra Adaptive Postfiltering (ESAP) algorithm to reduce the DCT blocking distortion. ESAP reduces the blocking distortion while preserving most of the image's edge information by adaptively postfiltering the decoded image using the block's spectral information already obtainable from each block's DCT coefficients. The algorithm iteratively applies a one dimensional log-sigmoid weighting function to the separable interpolated local block estimated spectra of the decoded image until it converges to the optimal PSNR with respect to the original using a 2-D steepest ascent search. Convergence is obtained in a few iterations for integer parameters. The optimal logsig parameters are transmitted to the decoder as a negligible byte of overhead data. A unique maxima is guaranteed due to the 2-D asymptotic exponential overshoot shape of the surface generated by the algorithm. ESAP is based on a DFT analysis of the DCT basis functions. It is implemented with pixel-by-pixel spatially adaptive separable FIR postfilters. PSNR objective improvements between 0.4 to 0.8 dB are shown together with their corresponding optimal PSNR adaptive postfiltered images.

* This work was supported by the NASA Goddard Space Flight Center Part-Time Graduate Study Program.

N95-14590

Synthetic Aperture Radar Signal Data Compression Using Block Adaptive Quantization

Gopinath Kuduvalli, Melanie Dutkiewicz, Ian Cumming[1]

Abstract

This paper describes the design and testing of an on-board SAR signal data compression algorithm for ESA's ENVISAT satellite. The Block Adaptive Quantization (BAQ) algorithm was selected, and optimized for the various operational modes of the ASAR instrument. A flexible BAQ scheme was developed which allows a selection of compression ratio/image quality trade-offs. Test results show the high quality of the SAR images processed from the reconstructed signal data, and the feasibility of on-board implementation using a single ASIC.

1 Introduction

Because of the growing volume of data collected in remote sensing satellites, the need for on-board data compression is increasing. This is particularly true of synthetic aperture radar (SAR) sensors, where swath widths and resolutions are limited by the on-board data handling capacity and the downlink bandwidth. Little use has been made of on-board data compression to date, because of the unavailability of signal processing capacity, power and weight constraints, reliability considerations, and the reluctance of users to accept any form of data degradation. However, the experience with the Magellan mission, and the progress of electronic technology has set the stage for the use of data compression in future operational SAR satellites.

Pioneering work on SAR data compression was done by JPL, and an algorithm called Block Adaptive Quantization (BAQ) was developed for the encoding of SAR signal (raw) data [1, 2]. The algorithm was first implemented on the Magellan mission to Venus [1], and later on the SIR-C mission. Following this, engineering studies were carried out by MacDonald Dettwiler for ESA, evaluating the Block Adaptive Quantization (BAQ), Vector Quantization (VQ) and Discrete Cosine Transform (DCT, JPEG version) data compression algorithms. Each algorithm had its advantages and its preferred application, and in the case of on-board compression of SAR signal data, BAQ was preferred, primarily because of its simplicity [3, 4]. There have been a number of other studies on SAR data compression in this period [5, 6, 7].

Following the initial demonstrations of feasibility, ESA funded a second project to design an ASIC for the next generation of European SAR satellites. The next planned SAR sensor after ERS-2 is the ASAR, to be flown on ENVISAT in the 1998 time frame. ASAR is to have a number of operating modes, each with its own engineering and application requirements. The modes varied from a 400 km wide-swath survey mode to a 100 Km precision imaging and calibration mode.

---

1. The authors are with MacDonald Dettwiler, 13800 Commerce Parkway, Richmond, Canada. Dr. Ian Cumming also holds the position of the MacDonald Dettwiler/NSERC Industrial Research Chair in Radar Remote Sensing at the University of British Columbia, Vancouver, Canada.

These different requirements pointed to the need for flexibility in data compression algorithms, where users could decide between the widest swath at moderate image quality and the highest precision with narrower swath widths.

Detailed requirements were placed on the encoder's signal/quantization noise ratio, preservation of statistics, radiometric linearity, phase error, spectral fidelity, discrete target accuracy and visual image quality. Additional requirements were placed on simplicity, reliability, scene-independence and real-time operation.

This paper describes the design and testing of a flexible BAQ SAR signal data compression algorithm to satisfy the above requirements. Section 2 summarizes the characteristics of SAR data, and the metrics selected to evaluate the effects of encoding. In Section 3, the selection and design of a flexible BAQ algorithm, including theoretical and experimental evaluation of several variants of the basic algorithm, is detailed. The hardware implementation of the algorithm in an ASIC is described in Section 4, and conclusions are given in Section 5.

2 SAR Data Characteristics and Evaluation Metrics

2.1 SAR Data Characteristics

SAR signal data is acquired by measuring the reflections of linear FM chirps transmitted and received with a SAR antenna. Thus SAR signal data consists of a two-dimensional convolution of the reflectances of a number of targets spread over the width of the linear FM chirp along range, and the SAR antenna beam width along the azimuth. Typically, this convolution operator is of the order of a few hundred samples in both range and azimuth directions [2]. The signal data are acquired in the complex domain by measuring both the in-phase and quadrature phase (I/Q) components of the received signal.

The convolution operation implicit in the acquisition of the SAR signal data results in a slow variation of the rms value of the signal in both range and azimuth directions. Further, this convolution operation results in a distribution of the received signal data that tends to be Gaussian [8, 9]. Thus SAR signal data can be modeled as Gaussian distributed random variable with a slowly varying rms value, and with little or no correlation between adjacent samples [1, 2]. Further, SAR signal data typically has a low signal to noise ratio — of the order of 10 to 15 dB. These characteristics govern the choice of a suitable algorithm for the compression of SAR signal data.

2.2 Evaluation Metrics

SAR data encoding takes place in the raw or signal data domain, whereas all the applications of SAR data are in the image domain. A convolution operator (matched filter) is used to transform the SAR signal data to the image domain. A consequence of the convolutional operator used to create the image is that there is no simple relationship between the properties of the data in the two domains. Thus to fully quantify and understand the effects of encoding, evaluation should take place in both the signal *and* image domains. In this way, one can gain insight into the cause, nature and severity of the signal domain error; into how the error is propagated into the image domain;

and, finally, into how the error might affect applications using the data. Evaluation metrics were chosen in order to:

- understand the *manner* in which the encoding error manifests itself,
- quantify the *severity* of the encoding error,
- understand the *mechanism* by which the encoding error is introduced.

The methods of evaluation which have been selected to meet these goals are [4]:

- **in the signal domain**: measurement of signal to quantization noise ratio (SQNR); analysis of effects of encoding on data statistics, data histograms, phase statistics and phase histograms,
- **in the image domain**: all the metrics used in the signal domain in addition to the analysis of effect of encoding on point target characteristics and spectral characteristics; measurement of radiometric linearity of encoding[1]; and measurement of global and local mis-registration effects.

3 Algorithm definition

3.1 Selection of candidate algorithms

Data compression algorithms generally exploit the correlation between samples of data to reduce the redundancy, and then apply a suitable quantization scheme to encode the resulting data. SAR signal data is best modeled as a Gaussian random variable with very low correlation between samples. Hence, the choice of a compression algorithm for SAR signal data reduces to that of selection of a suitable quantizer. The fundamental idea behind the block adaptive quantization (BAQ) is to adaptively vary the step sizes of a non-uniform quantizer based on the estimated variance of a block of samples [1, 2]. This achieves a wider overall dynamic range at the quantizer output, for the same number of quantization levels, than simple uniform quantization of the data. Several variants of this basic idea are possible, based on the choice of the quantizer.

In this study, the design of a compression algorithm with flexible compression ratios was approached in two stages. The first stage of the study was to select the best form of the BAQ algorithm, identify important parameters and determine their optimum values using experimental evaluation with actual SAR signal data. The second stage of the study was to extend the selected version of the algorithm for flexible compression ratios, evaluate the algorithm at different encoding rates, and fully specify the design of the algorithm for an ASIC implementation.

The variants of the BAQ algorithm selected as potential candidates for implementation are described in the following subsections.

---

1. Radiometric linearity is a measure of how well the algorithm preserves the intensity levels of homogeneous regions within the image. Linearity is determined by plotting mean intensity of homogeneous regions (ranging from dark to very bright) in the decoded image versus mean intensity of the same regions in the original image. Perfect linearity would give an exact fit to a straight line with slope of 1.0 and zero offset.

3.1.1 Block Adaptive Quantization (BAQ):

This scheme is based on JPL's BAQ implementation for the Magellan mission [1]. The absolute values of I and Q are compared with a threshold derived from a block of input signal data samples, and encoded with 1 bit. The sign bit of the I and Q samples constitute the second bit. The threshold and reconstruction levels are chosen to result in minimum mean square quantization error for 2-bit quantization of a Gaussian random variable with a variance equal to the sample variance of the block.

This idea can be extended to provide greater compression accuracy by increasing the number of thresholds and allowing more bits per codeword. Three-bit BAQ requires 3 thresholds, and 4-bit BAQ requires 7 thresholds. The quantizer for 3 and 4 bits consist of successively comparing the absolute values of I and Q with the set of thresholds computed from a block of samples, and encoding the result of comparison with a 2 or 3-bit codes; the sign bit constitutes the additional bit.

3.1.2 Block Adaptive Magnitude Phase Quantization (BMPQ):

In BMPQ, the input I/Q values are transformed to magnitude-phase representation. The phase component is uniformly distributed and the magnitude is Rayleigh distributed. The quantization thresholds and reconstruction levels are determined for each component to minimize the mean square quantization error for the respective distributions. The number of bits allocated to the magnitude and phase components for quantization are varied to achieve the best overall performance. Table 3-1 gives the theoretical performance of the quantizer for different bit allocations to magnitude and phase.

Table 3-1 SQNR performance of BMPQ for different bit allocations

Number of bits/sample for encoding the magnitude	Number of bits/sample allocated for encoding phase				
	1	2	3	4	5
0				6.48	6.63
1			9.19	10.70	11.19
2		6.63	11.41	14.57	15.93
3	1.38	6.89	12.44	17.21	20.22
4	1.38	6.98	12.79	18.40	23.11
5	1.39	7.00	12.89	18.78	24.38

The cross diagonals of Table 3-1 represent the SQNR for a constant encoding rate or compression ratio. The shaded cells highlight the bit allocation combination which results in the best performance for the given number of bits per sample. For example, at 2 bits/sample (i.e., 4 bits per

complex sample) the highest SQNR is expected with 1 bit allocated to magnitude and 3 bits for phase.

3.1.3 Block Adaptive Histogram Equalization Quantization (BHEQ):

BHEQ consists of transforming the I/Q samples from Gaussian distribution to uniform distribution using the block rms value. This operation is recognized as the classical histogram equalization, with the added feature that the histogram is known a-priori. The transformation consists of computing the cumulative distribution function of the Gaussian distribution, and can be performed using look-up tables. The resulting 8-bit transformed I/Q values can be quantized to the required number of bits simply by truncation.

BHEQ minimizes the quantization error in the histogram-equalized domain. This is *not* equivalent to minimizing the quantization error in the original signal with Gaussian distribution. Thus BHEQ results in lower SQNR than BAQ at all encoding rates. The main reason for studying this type of quantizer is that the quantizer is essentially identical for different compression ratios.

3.1.4 Block Adaptive Complex Quantization (BACQ):

BACQ consists of treating a pair of I and Q values as a complex sample, and designing a generalized complex quantizer using quantization boundaries and reconstruction levels in the two-dimensional (2-D) space. Straight forward implementation of 2-D quantizers using look-up tables require large amount of memory and precludes on-board hardware implementation. However, the approach used in the case of BHEQ can be used to bring down the size of the look-up tables to more manageable levels. The I/Q samples are converted to uniform distribution, as in the case of BHEQ, using look-up tables. A second look-up table is used to quantize the transformed I/Q values into a single complex quantizer code.

A possible selection of quantizer reconstruction levels and the corresponding optimal quantization boundaries in the 2-D space is shown in Figure 3-1.

3.2 Evaluation of BAQ Variants

Table 3-2 gives the theoretical (signal domain) SQNR performance of the four variants of the BAQ algorithm. Previous studies have shown that the SQNR is the most significant signal domain parameter that affects the image domain performance of a quantizer for SAR signal data compression [3]. Table 3-2 shows that the expected SQNR performance of the four candidate algorithms are very close to one another, with BAQ outperforming the other algorithms by a slight margin. (The shaded cells highlight the best performing algorithm at each encoding rate.) Simulations showed that all the four variants maintain these performance levels over a dynamic range of 40 dB, for 8-bit data [10].

Analysis showed that the best signal domain *phase* performance is achieved by BMPQ and BACQ. This is because BMPQ and BACQ have more reconstruction levels for phase for a given number

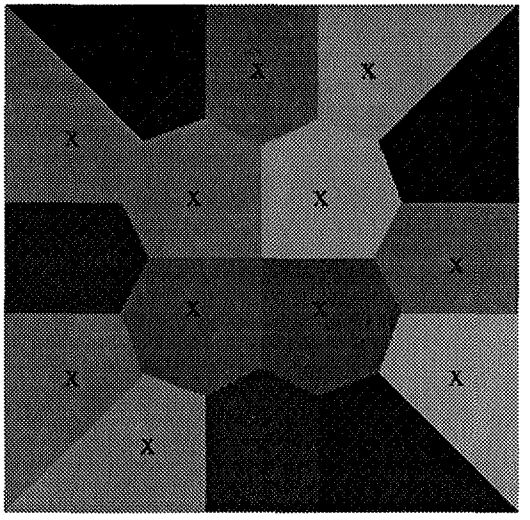


Figure 3-1 Quantizer reconstruction levels and quantization boundaries for a 2-D quantizer

Table 3-2 SQNR performance of the variants of BAQ

bits/sample	SQNR in dB			
	BAQ	BMPQ	BHEQ	BACQ
2	9.30	9.19	9.15	9.15
3	14.62	14.57	14.34	N/E[a]
4	20.22	20.22	19.94	N/E[a]

a. Not evaluated

of bits for encoding. However, whether this could result in any improvement in the performance in the processed image domain could only be verified with experimental evaluation.

Compression at 2 bits/sample was selected as a baseline for comparison of the performance of the variants of the BAQ algorithm using simulations with actual SAR signal data. Experimental evaluation of the four BAQ variants at 2-bits per sample showed that:

- The SQNR performance of the four variants was within 0.7 dB of each other, with BAQ giving the best performance of the four variants. For the *detected image*, an average SQNR of about 14 dB was achieved in all cases.
- It had been conjectured that using M/P representation might result in improvement in encoding performance. The results showed, however, that although BMPQ did have the best phase performance in the signal domain, the lower signal domain SQNR of the individual I and Q components prevented this result from being propagated into the image domain. Among the various bit allocation possibilities for BMPQ, only BMPQ(1,3) — i.e. 1 bit

allocated to magnitude, 3 bits to phase — was comparable in performance to the other three variants.

- All the variants showed very good visual image quality, good fidelity in preserving data magnitude and phase distributions, and produced no mis-registration effects.
- The spectra of the detected images were virtually indistinguishable from that of the original image for all variants.
- Apart from a small loss in total peak energy, the point target characteristics for all variants were very well preserved, with negligible distortion in peak phase, 3 dB widths in range and azimuth, peak or integrated sidelobe ratios for all variants.
- Radiometric linearity was perturbed least by BAQ and most by BACQ.
- The phase performance of all the four variants, when encoding to 2-bits/sample, were found to be below acceptable levels for certain specialized applications. An rms phase error of about 30^{o} was found in the reconstructed processed image data. The rms value of phase error weighted by the magnitude was about 15^{o}. This is thought to be outside the limits of acceptability in applications such as SAR interferometry — an rms weighted phase error of less than 10^{o} is desired for such applications.

3.3 Flexible BAQ algorithm (FBAQ)

The intial study of different variants of BAQ established that the overall performance in both the complex image domain and the detected image domain was very similar for all the four variants. BAQ performed slightly better in terms of SQNR. In the case of BMPQ and BACQ, although somewhat better signal domain phase performance was observed, it did not translate to an improved image domain SQNR or phase performance.

For hardware implementation with flexible compression ratios, BHEQ is the most straight forward since it involves no additional hardware for extension from single compression ratio to flexible compression ratios. However, BHEQ requires a large amount of memory to implement the look-up table for the histogram equalizing transformation. Further, the look-up table has to be accessed once for every I or Q sample for encoding. This is a serious limitation for on-board implementation at high data rates.

BAQ requires a total of 11 different look-up tables to achieve flexible compression ratios at 2, 3, and 4 bits/sample. Further, the encoder requires a successive comparator which is a little more complex than the simple truncation involved in the case of BHEQ. However, the look-up tables need to be accessed only once for every block, thus simplifying the design of the look-up tables and their addressing in hardware.

BMPQ, which involves rectangular to polar conversion in hardware, requires higher hardware complexity than both BAQ and BHEQ. The 2-D quantizer for BACQ is inherently limited to low bit rate encoding.

With these considerations, BAQ was selected as the most appropriate variant for implementation as an on-board SAR data encoding algorithm with flexible compression ratios. We have called this

extension of the BAQ algorithm to incorporate flexible compression ratios the Flexible BAQ algorithm (FBAQ).

3.4 Optimal of Selection of Implementation Parameters

A number of parameters were identified for the optimal implementation of the FBAQ algorithm for on-board use. Experimental evaluations were performed at 2 bits/sample, since the optimal selection of these parameters were deemed independent of the compression ratio selected.

- The size and shape of the block of samples from which to estimate the optimal thresholds for encoding depends upon the nature of variation the rms value of the SAR signal data along the range and azimuth directions. Experiments showed that the BAQ algorithm is not sensitive to the changes in the block size in the range of ~64 to ~512 samples. Further, the use of two-dimensional blocks did not result in any significant improvement in the performance of the algorithm. For hardware simplicity, and to limit encoding delay, a one-dimensional block oriented along range is preferred.
- Sub-sampling of the block, and using thresholds computed from the statistics of the previous block were considered to simplify the on-board implementation. It was however observed that both these options result in a small degradation of performance of the algorithm. Preliminary hardware analysis showed that these simplifications were not required.
- Independent encoding of I and Q channels was considered to reduce the effect of gain and offset imbalance between channels in the on-board sensor. The effect of this imbalance on the performance of the quantizers was found to be minimal. It was concluded that the effective doubling of complexity of the hardware required for the independent encoding of I and Q channels is not desirable.

Based on the results of these experimental evaluation, the final the set of the parameters for on-board implementation of FBAQ were chosen as shown in Table 3-3.

Table 3-3 FBAQ implementation parameters

encoded bits/sample	Block size along range (Block size along azimuth=1)	LUT size
2 bits	126 pairs of I/Q samples	64x1, 7-bit thresholds
3 bits	84 pairs of I/Q samples	64x3, 7-bit thresholds
4 bits	63 pairs of I/Q samples	256x7, 7-bit thresholds

Note that if the quantizers are linearly spaced across the dynamic range for 8-bit data, the optimum number of entries per threshold look-up table (LUT) is 256. For address space considerations in the on-board implementation, a total look-up table size of 2K entries was preferred. As a result, a slightly sub-optimal size of look-up table is used for encoding at 2 and 3 bits/sample[1]. However, this does not affect the performance significantly.

The thresholds for each block were determined by estimating the rms value using from all samples in the current block, as opposed to a subset of the samples, as was used in the Magellan BAQ implementation [1]. This requires that all the samples of each block have to be stored in a buffer memory until the thresholds for that block become available. This additional memory was determined to result in negligible increase in hardware complexity.

3.5 Evaluation of FBAQ algorithm

A complete performance evaluation of the FBAQ algorithm was conducted by running end-to-end tests at each of the three available bit rates. Each end-to-end test consisted of the following steps:

- raw data encoding and decoding,
- signal domain evaluation,
- SAR processing of original and decoded data sets,
- processed image domain evaluation.

Three data sets with a variety of scene content were used during the test campaign — an agricultural scene from Flevoland, Holland, which included coastline, inland sea, fields and SAR transponders; a mountainous region of Sardegna, Italy; and a suburban region of Flevoland, Holland, which included an airfield and buildings. The latter data set was taken at far range, and was included to test the algorithm under low scene SNR conditions.

Table 3-4 SQNR and Phase Performance Ranges of FBAQ Algorithm

Parameter	2 bits	3 bits	4 bits
Signal domain SQNR, magnitude (dB)	11.10 - 11.64	15.55 - 16.84	21.65 - 22.89
rms phase error (deg)	18.09 - 18.11	11.20 - 11.42	6.92 - 7.00
mean abs. phase error (deg)	14.03 - 14.08	7.87 - 8.05	4.47 - 4.50
Image domain SQNR, magnitude (dB)	14.14 - 14.68	19.29 - 20.16	25.12 - 25.96
rms phase error (deg)	29.78 - 34.61	17.56 - 21.71	10.00 - 12.48
rms weighted phase error (deg)	14.06 - 17.18	7.27 - 9.37	3.60 - 4.49

Table 3-4 shows the range of performance results for the FBAQ algorithm obtained using the three data sets at all three bit rates. The results of the evaluation showed that:

- the images from compressed data had excellent visual quality at all three bit rates, being virtually indistinguishable from the original image, except for a slight increase in background noise at 2-bits/sample. Figure 3-2 and Figure 3-3 show the Original, reconstructed and error images for the Flevoland data set. Note that the error images have been multiplied by a factor of 10 - no structure is visible at x1 magnification.

1. It should be noted that if the number of quantizers is reduced, log spacing gives better performance at low powers and linear spacing gives better performance at high powers. Log spacing does however considerably increase the addressing complexity.

- rms weighted phase error is in the range 14^{o} - 17^{o} at 2-bits/sample, 7^{o} - 9^{o} at 3-bits/sample and of the order of 4^{o} at 4-bits/sample. An rms weighted phase error of 10^{o} or less should be acceptable for SAR applications requiring high phase integrity, encoding at both 3- or 4-bits/sample meets this requirement.
- the statistical moments are slightly degraded at the lowest bit-rate (2-bits/sample) but no significant degradation was observed at either 3- and 4-bits/sample,
- image data and phase distributions are well reproduced at all bit rates,
- point target characteristics are well reproduced at all bit rates, with the only noticeable effect being a small loss in total peak energy at 2-bits/sample,
- the spectra of the detected images were virtually indistinguishable form those of the original image for all bit rates.
- no mis-registration was observed at any of the bit-rates,
- radiometric linearity was slightly degraded at 2-bits/sample, but excellent at 3- and 4-bits/sample,
- the algorithm performance is relatively insensitive to scene content and hence no reprogramming of threshold look-up tables is required for the algorithm as the characteristics of the scene under view changes,
- the algorithm is effective on far- as well as near-range data, with only a slight increase in SQNR and phase error observed at far range.

Thus this algorithm has been found to result in images which meet the requirements of applications dependent on visual properties of the image at all three bit-rates - with the lowest bit-rate giving the additional benefit of allowing wider swath width coverage for the same transmission bit-rate - and to meet the requirements of applications requiring good radiometric and phase performance at 3- and 4-bits/sample.

4 Implementation for On-Board Use

The preliminary designs of the ASAR on-board data handling system were studied, and it was determined that the data compression scheme could be implemented by a single ASIC placed between the A/D converter and the main data handling memory. In addition to the selection of 2, 3 or 4-bits per sample, the ASIC could be programmed to pass 8-bit data through without encoding to perform built-in self tests.

A block diagram of the ASIC functionality is shown in Figure 4-1. A range line of up to 6000 complex samples is divided into blocks of 63 to 126 samples and the rms value of each block is estimated by accumulating the absolute values of the I and Q portions of the complex SAR signal data. This is done with the full 8-bit precision of the A/D converter. The rms estimate is used to select a set of thresholds, depending upon whether 2, 3 or 4-bits per sample are selected. The thresholds are used to quantize the samples in the same data block as the estimate was taken. A successive comparator approach was selected as the most efficient for the ASIC implementation. The index of the selected threshold is multiplexed into the encoded data block.

The threshold values are stored in a PROM outside of the encoder chip. Although these can be reprogrammed, in has been determined in tests that the FBAQ scheme is sufficiently general that there is no need to change threshold levels when the scene content changes.

A synthesizable VHDL model of the FBAQ algorithm has been developed using the V-system VHDL compiler and simulator running under Windows on a PC. The interfaces of the ASIC have been designed to fit into the ASAR Data Subsystem. The ASIC design has been verified using both internal test vectors and real SAR data. In the latter case, the ASIC output was compared with the output of the simulation used in the algorithm study.

ABB HAFO and Matra MHS have been selected as foundries for the chip, and the manufactured ASIC is expected to have the specifications shown in Table 4-1..

Table 4-1 Preliminary specifications of the FBAQ ASIC

Technology	0.8μm CMOS
Estimated Gate Count	< 15,000
Maximum Operating Frequency	20 MHz
Radiation Tolerance	> 30 kRad
Power Dissipation	< 1 w
Packaging	84-pin Quad Flat Pack

5 Conclusions

After assessing the user requirements for satellite SAR image quality, and the image quality/ coverage trade-offs in sensor deployment, it was concluded that an operational ASIC should have the flexibility of encoding to a user-selectable variable precision of 2, 3 and 4-bits per sample. The project objectives were to design the algorithm, to evaluate the accuracy and to design the ASIC for such a requirement. The desired results were obtained in that the 3-bit case was found to yield very good image quality and was deemed suitable for most users. However, users who wanted a very large swath width with reduced emphasis on image quality could choose the 2-bit option, and users who had very precise < image quality requirements could select the 4-bit option (which gives image quality almost identical to the full 8-bit case).

Thus the project has shown that a flexible on-board data compression scheme can be designed for SAR signal data which gives significant compression ratios without an appreciable degradation in image quality. The scheme has been implemented in a single ASIC, whose simplicity, reliability, flexibility and low power consumption make it suitable for use on-board a remote sensing satellite. A prototype ASIC is now being manufactured for the ASAR breadboard with the intent of incorporating it in the ENVISAT data handling system.

Looking beyond the ENVISAT program, the FBAQ encoder is expected to yield additional satellite SAR system improvements. One example is increased range bandwidth, which will give a direct improvement in SAR image quality. Once additional power is available to drive the SAR

power amplifiers, the FBAQ algorithm will allow a doubling of range bandwidth, keeping the swath width and data rates the same as on current missions.

6 Acknowledgments

The research work was carried out under contract to the European Space Agency, with Mr. J.-L. Marchand and Mr. D. Chaturvedi as Technical Officers. The detailed ASIC design and testing was performed by Mr. Peter Roos of Saab Ericsson Space, Gothenburg, Sweden.

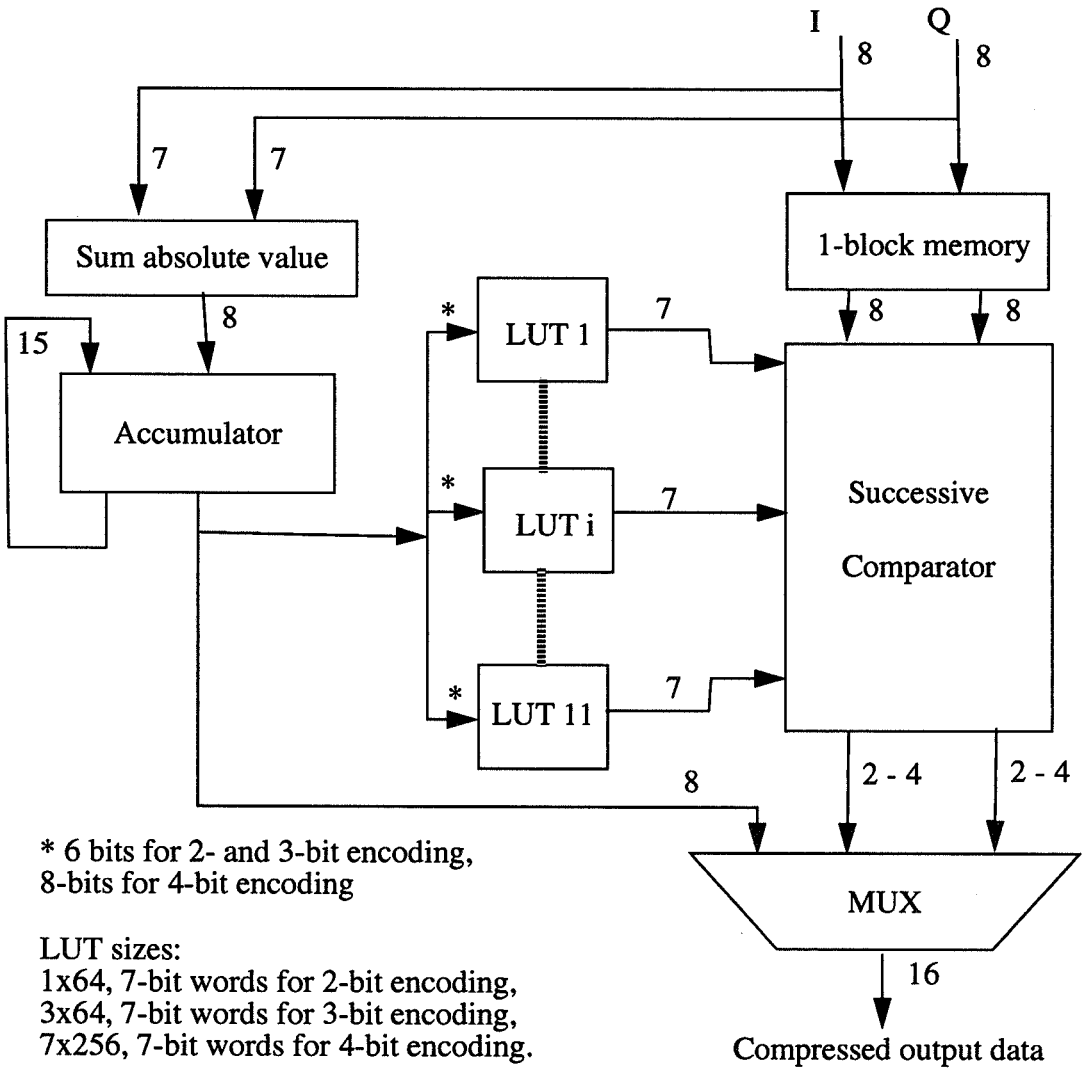


Figure 4-1 Schematic Diagram of FBAQ Encoder with Table Sizes and Word Lengths

7 References

1 *Block Adaptive Quantization of Magellan SAR Data*, R.Kwok and W.Johnson, IEEE Trans. Geoscience and Remote Sensing. Vol. 26, No. 5, pp. 375-383, July 1989.

2 *Synthetic Aperture Radar - Systems and Signal Processing*, J. C. Curlander and R. N. McDonough, John Wiley and Sons Inc., 1991.

3 *Data Encoding Techniques Study, Final Report* under Contract No. 9122/90/NL/PR(SC), conducted for ESTEC by MacDonald Dettwiler, April 1992.

4 *Methods of Evaluating the Effects of Encoding on SAR Data*, M.Dutkiewicz and I.Cumming, Proceedings of 1992 NASA Space and Earth Sciences Data Compression Workshop, Snowbird, Utah, 1992.

5 *Vector Quantization Used to Reduce SAR Data Rates,* T. Gioutsos, SPIE Millimeter Wave and Synthetic Aperture Radar, Vol. 1101, pp. 116-128, 1989.

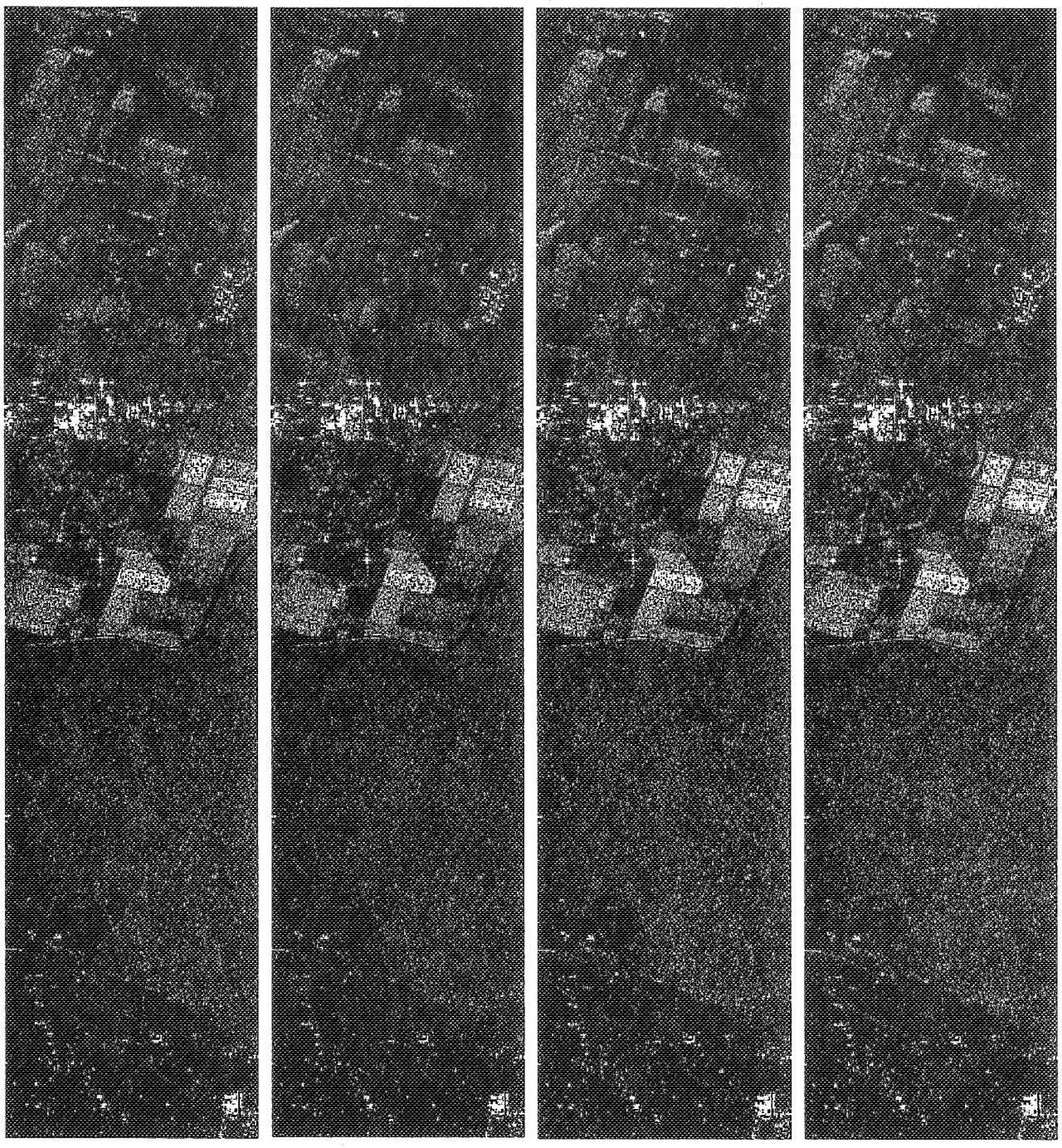
6 *Fusion of Block Adaptive and Vector Quantizer for Efficient SAR Data Compression,* A. Moreira and F. Blaser, pp. 1583-1585, IGARSS 1993.

7 *A Comparative Study of SAR Data Compression Schemes,* C. Lambert-Nebout, D. Massonnet, and B. Rogron, Proceedings of the IEEE Data Compression Conference, pp. 467, Snowbird, Utah, March, 1994.

8 *Statistical properties of speckle patterns,* J.W. Goodman in *Laser Speckle and Related Phenomena,* Topics in Applied Physics, J. Dainty Ed., Vol. 9, Springer Verlag, NY, pp. 9-75, 1975.

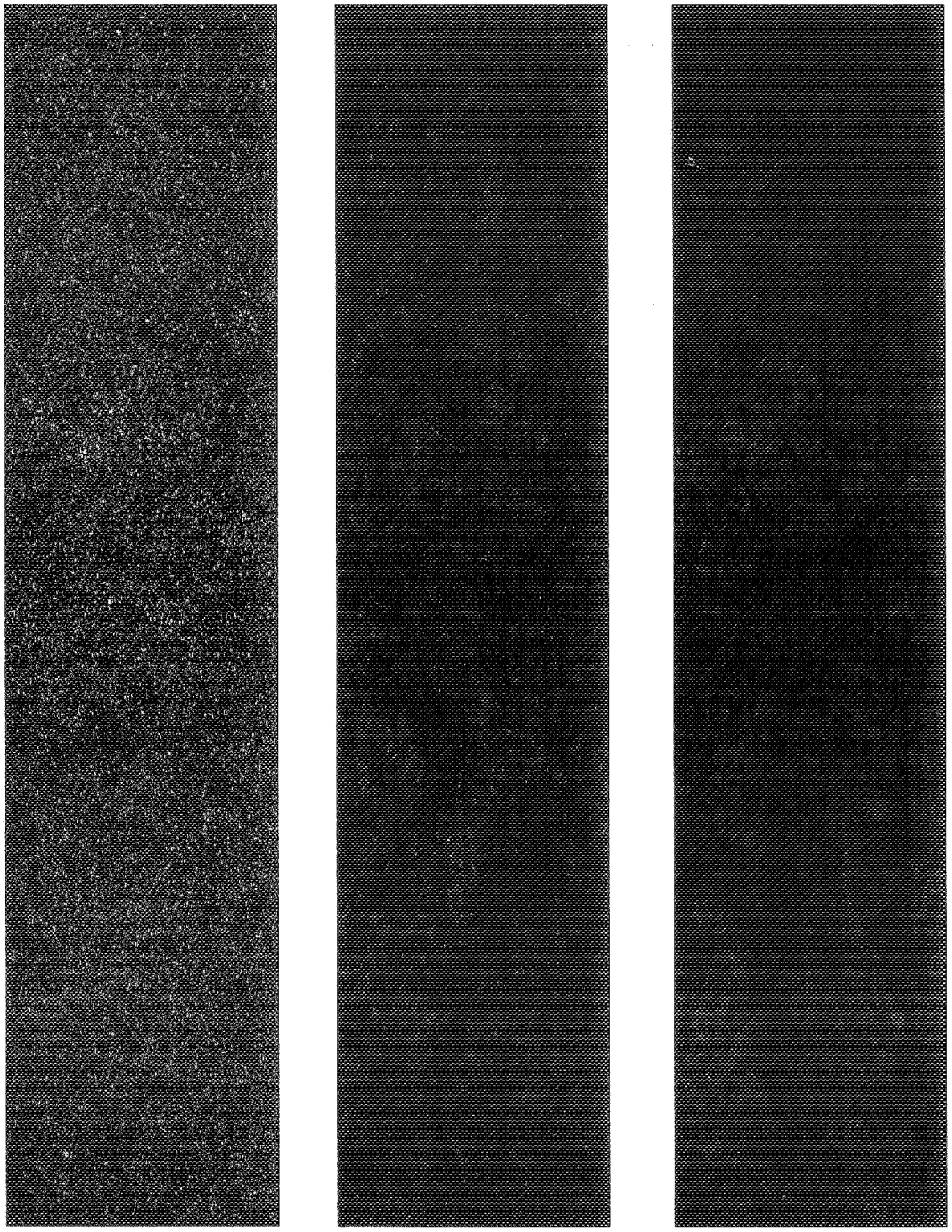
9 *Microwave Remote Sensing, Active and Passive,* Vol.2, NY, Artech, 1986.

10 *SAR Pre-Processing On-Board, Interim Report on the Algorithm Study,* under Contract No. 10135/92/NL/FM, conducted for ESTEC by MacDonald Dettwiler, May 1993.



Original 2-bits/sample 3-bits/sample 4-bits/sample

Flevoland Images for Original and Decoded Data



2-bits/sample 3-bits/sample 4-bits/sample

Flevoland Difference Images x10

N95-14591

Compression of Regions in the Global Advanced Very High Resolution Radiometer 1-Km Data Set

Barbara L. Kess[1]
University of Nebraska - Lincoln
Lincoln, Nebraska 68588-0115
bkess@cse.unl.edu

Daniel R. Steinwand[2]
EROS Data Center
Sioux Fall, South Dakota 57198
stein@suno.cr.usgs.gov

Stephen E. Reichenbach
University of Nebraska - Lincoln
Lincoln, Nebraska 68588-0115
reich@ser.unl.edu

ABSTRACT

The global advanced very high resolution radiometer (AVHRR) 1-km dat set is a 10-band image produced at USGS' EROS Data Center for the study of the world's land surfaces. The image contains masked regions for non-land areas which are identical in each band but vary between data sets. They comprise over 75 percent of this 9.7 gigabyte image. A quad tree is used to find and compress boundaries for land and masked regions. The mask is compressed once and stored separately from the land data which is compressed for each of the 10 bands. The mask is stored in a hierarchical format for multi-resolution decompression of geographic subwindows of the image. The land for each band is compressed by modifying the method described in Kess, Steinwand and Reichenbach (1994) to ignore fill values. This multi-spectral region compression efficiently compresses the region data and precludes fill values from interfering with land compression statistics. Results show that the masked regions in a one-byte test image (6.5 Gigabytes) compress to .2 percent of the 557,756,146 bytes they occupy in the original image, resulting in a compression ratio of 89.9 percent for the entire image.

1. INTRODUCTION

The Global Advanced Very High Resolution Radiometer (AVHRR) 1-km project is an example of the need for data compression in the Earth Observing System Distributed Information System (EOSDIS). As part of this project, the U.S. Geological Survey's (USGS) Earth Resources Observation Systems (EROS) Data Center, in conjunction with other international data centers and science groups, is planning to produce global data sets at 1-km resolution, one data set per 10-day period. This data set contains just less than 10 gigabytes of data. Without any compression, the data set requires at least 15 CD-ROMs that hold 660 MB each. The requirements for compression of this data set include lossless decompression of geographic subwindows of the data at multiple resolutions. Compression methods that divide the image into blocks and compress each block with a hierarchical format that allows multiresolution decompression have been developed (Kess, Steinwand, and Reichenbach, 1994).

Since the purpose of this data set is for study of the world's land surfaces, all non-land regions are masked and set to a constant. Mask values are used to fill regions of water, unused parts of the framed data in the map projection, and land where there is no data. The masked regions are exactly the same in all 10 bands of the image, but may vary between data sets. They

---

[1] Hughes STX Corporation. Work performed under consultant agreement no. 93-9002-I1904.
[2] Hughes STX Corporation. Work performed under U.S. Geological Survey contract 1434-92-C-40004.

comprise at least 75 percent of the image, making efficient compression of the fill areas a major factor in the success of the compression algorithm. A quad tree is used to describe and compress boundaries for the masked regions and land regions. Since the masked regions are exactly the same in all ten bands of the image, it is only necessary to compress the region data once for the entire image, rather than once for each band. The mask data is stored separately from the land data and is accessible during decompression of each of the 10 bands. The following describes the approach used for separating the region data from the land data. Results are given for a 10-band test image containing one-byte data in all 10 bands, at 6.5 Gigabytes. The full 9.7 Gigabyte image (which contains 5 bands with two-byte integer pixels) was not yet available for our test purposes.

2. COMPRESSION OF REGIONS

The quad tree used to compress the region boundaries is a region quad tree as described in Samet (1984). This 4-way tree structure represents a recursive decomposition of the image into quadrants. When each of the four child quadrants of a parent node are found to be homogeneous, the parent node is used to represent the information present in all four child quadrants. In the following example, solid regions are shown with black nodes and non-solid regions are shown with white nodes. Black nodes that are close to the root of the tree represent large solid regions of data (See fig. 1).

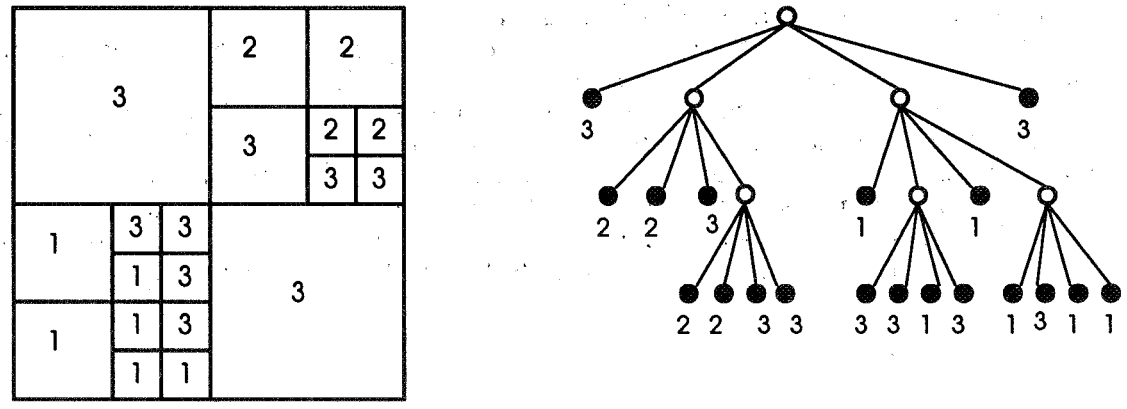


Figure 1. Regions of an 8x8 block and its respective quad tree.

The levels of the quad tree can be easily used to store data for resolution levels that differ by a factor of four. Each internal node stores a subsampled value from its four children. The subsampling method chooses the upper left pixel in each 2 *x* 2 block, which means that each internal node in the quad tree receives the value of the first child node.

The mask compression algorithm initializes each leaf node with the value of the pixel it represents. All leaf nodes that represent land pixels receive a constant that represents land regions. Thus, each leaf node is marked as being part of one of the four possible regions: water, land, land with no data, and unused parts in the framed map projection.

The tree is built from the leaf nodes up to the root, giving each parent the value of its first child and setting each internal node's solid flag to true if all four children are solid and have the same value. Blocks of size $2^n \times 2^n$ require $2^{2n}+2^{2n}/3$ nodes to build the tree. The testing was done with a block size of *128 x 128*, which requires *21,845* nodes. This makes it feasible to build and store the tree in memory during compression and decompression (See fig. 2).

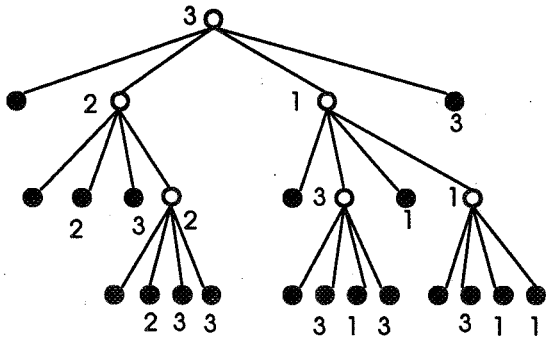


Figure 2: Quad tree from Figure 1 with value in first child promoted to parent mode.

To compress the tree a breadth first search is done, starting at the root. Each node sends a maximum of 3 bits to the output stream. The first bit specifies whether the node is solid (or not solid) and two more bits are used to give the value represented by the node. A queue is used to determine the visiting order for the nodes. If a node is not solid, it enqueues each of its children. If a node is solid, none of its children are enqueued because all necessary information for reconstructing its children has already been given to the output stream. If a node is a first child it does not send its value to the output because its parent's value has already been compressed. If a node is a leaf, it does not send a solid bit because it has no children and only represents one value. The worst scenario with this tree is that there are no solid regions in which case two bits are transmitted for each sample in the original block of data, plus 1 bit to designate that each internal node is not solid. If n is the number of samples in the original image then the maximum number of bits used for the compressed data is $2n + n/3$. The compressed bit stream is shown in Figure 3. Each row represents the bits used to compress a level of the tree which was shown in Figure 2.

```
011
1 010 001 111
1 110 111 010 1 011 101 001
10 11 11 11 01 11 11 01 01
```

Figure 3: Compressed bit stream for quad tree in Figure 2.

3. LAND COMPRESSION

The land compression algorithm compresses each block with a hierarchical method proposed by Sloan and Tanimoto (1979). The pixels are reordered by placing pixels needed for the coarsest resolution at the beginning of the block, followed by pixels needed to fill in the next resolution, until the full resolution image is restored losslessly. The block is then de-correlated with a JPEG prediction scheme that predicts each pixel based on the value of the previous pixel (Wallace, 1993). The decorrelated data is coded with Huffman coding (Huffman, 1962).

Since the region data is already compressed, the land compression algorithm needs only to compress land data. Some blocks, however, contain a mixture of land data and fill data. In these cases, the land compression algorithm is modified to ignore the fill data. During de-correlation and coding, each pixel is tested to determine if it is a land value or a mask value. All mask values are ignored during decorrelation and coding, producing compressed data that contains only land values. This improves the compression statistics for each band because no

extra space is given to fill data and the presence of fill data does not affect statistics used to compress the land data.

4. MASK DECOMPRESSION

Decompression of each block compressed with the mask separation approach involves first decompressing the mask and then filling in the land values where they belong. Prior to decompression of specific blocks, the quad tree is created in precisely the same manner as during compression. If the user specifies a resolution other than 1 km for the decompressed image, then the number of leaf nodes is computed to match the number of pixels in the decompressed image. Since all blocks are decompressed to the same resolution, the quad tree has the correct size for each block. The algorithm finishes when it has traversed all of the leaf nodes, so it automatically decompresses each block to the correct resolution. Each leaf receives the offset into the decompressed block where its pixel value belongs. During compression, pixels were transferred from the image to the leaf nodes of the tree. Now, during decompression, the pixel values are transferred from the leaf nodes to their appropriate offset in the decompressed block.

Once the tree is created, the compressed information is read and used to fill in the nodes of the tree. Every internal node enqueues its four children into the queue to be visited. Each node, except for the root node, checks the solid flag in its parent. If its parent is solid, then it simply inherits the parents solid flag and node value. If the parent node is not solid, then the child node receives bits from the compressed data, 1 bit for the solid flag and 2 bits for the value. Some exceptions to this are that the first child inherits the value from its parent rather than retrieving it from the compressed data, and leaf nodes do not retrieve a solid bit. After all nodes of the quad tree have been visited, the leaf values are ready to be copied to the correct offset in the decompressed block. A constant value is placed into each pixel that requires a land value.

5. LAND DECOMPRESSION

After the mask decompression routine has stored region data in the decompressed block, the land decompression routine decompresses the land values and places them into pixels that contain a constant value, representing land. For each land value decoded from the compressed input, the algorithm computes its offset into the decompressed block. If this position contains a mask value, the algorithm moves to the next position. Decompressed samples are only allowed to be copied to pixels that contain a land constant.

6. BLOCKS OF UNEVEN SIZE

The global image dimensions are not evenly divisible by 2^n. The compression algorithm is designed for blocks whose dimensions are $2^n \times 2^n$. This leaves blocks on the right and bottom sides of the image that are not full. The mask compression and decompression algorithm accommodates these blocks by adding pad values to the tree. If a leaf value falls into the padded area it is noted as such and ignored when the values are assigned to internal nodes of the quad tree. This maintains the integrity of the quad tree, but does not send any bits to the compressed output for the padded pixels. When the tree is recreated during decompression it knows exactly which leaf nodes fall into the padded areas and ignores them when copying leaf values to the decompressed block.

The land compression and decompression algorithm does not depend on a $2^n \times 2^n$ block size. Changing the block size does not affect the land algorithm's ability to reorder the data during compression and to put the data back into the correct position during decompression.

7. RESULTS

Data from a 10-band image with one byte data in each band was compressedwith the hybrid approach described in Kess, Steinwand, and Reichenbach (1994) and also with the mask separation approach described in this paper. The hybrid approach compresses blocks that contain only two or three distinct values with run length encoding and it compresses solid blocks with two bytes. The other blocks are compressed with the land compression algorithm. The header bytes are used to store the block table and a global Huffman table. The number of bytes out is the actual space required for the data in the image.

Compression with Hybrid Approach

Band	Bytes In	Bytes Out	Header Bytes
1	694,417,757	100,498,557	172,352
2	694,417,757	108,417,401	172,352
3	694,417,757	91,753,214	172,352
4	694,417,757	94,351,654	172,352
5	694,417,757	94,554,288	172,352
6	694,417,757	83,834,431	172,352
7	694,417,757	71,682,592	172,352
8	694,417,757	56,498,487	172,352
9	694,417,757	56,875,854	172,352
10	694,417,757	58,676,725	172,352
TOTAL	6,944,177,570	817,143,203	1,723,520
Total Compressed Size:		818,866,723	bytes
		780.932	megabytes
Compression Ratio:		88.21%	

The mask separation approach uses the approach described in this paper in which region data is compressed separately from the land data. In the results using the mask separation approach, the number of bytes out for each band is the amount of space required to compress the land data using a JPEG prediction scheme and global Huffman coding. Preliminary results using an adaptive Huffman algorithm (not reported here) instead of global Huffman coding indicate at least a 20 megabyte improvement for the entire image.

Compression with Mask Separation Approach

Band	Bytes In	Bytes Out	Header Bytes
Mask	694,417,757	992,345	170,304
1	694,417,757	88,893,680	171,328
2	694,417,757	96,178,846	171,328
3	694,417,757	77,461,785	171,328

4	694,417,757	78,718,869	171,328
5	694,417,757	79,199,998	171,328
6	694,417,757	68,360,373	171,328
7	694,417,757	65,899,562	171,328
8	694,417,757	45,091,220	171,328
9	694,417,757	49,054,680	171,328
10	694,417,757	47,687,610	171,328
TOTAL	6,944,177,570	697,538,968	1,883,584

Total Compressed Size:	699,422,522	bytes
	667.021	Megabytes
Compression Ratio:	89.93%	
Improvement of Mask Separation to Hybrid Approach:	14.59%	

Distribution of Pixels in Each Band

Pixel Type	Bytes	% of Image
Unused portions	177,245,765	25.52%
Water	368,222,266	53.03%
Land without data	12,288,115	1.77%
Land	136,661,611	19.68%
Total	694,417,757	100.00%
Total Mask Bytes:	557,756,146	
Mask Compression Ratio:	99.82%	

8. REFERENCES

D. A. Huffman, 1962, "A Method for the Construction of Minimum Redundancy Codes", *Proceedings IRE*, v. 40, 1962, pp. 1098-1101.

B. L. Kess, D. R. Steinwand, and S. E. Reichenbach, 1994, "Compression of the Global Advanced Very High Resolution Radiometer 1-Km Data Set", in *International Geoscience and Remote Sensing Symposium Proceedings*, August 1994.

W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*, Van Nostrand Reinhold, New York, 1993.

H. Samet, "The Quadtree and Related Hierarchical Data Structures", *Computing Surveys*, v. 16, no. 2, June 1984.

K. T. Sloan and S. L. Tanimoto, "Progressive Refinement of Raster Images", *IEEE Transactions on Computers*, November 1979, v. C-28, no. 11, pp. 871-4.

Lossless Compression of NOAA-AVHRR Satellite Data

Seishi Takamura and Mikio Takagi
Institute of Industrial Science, University of Tokyo

Abstract

A high-performance lossless compression system for satellite NOAA data is developed. The data is called "high resolution picture transmission" (HRPT) data, and consists of around 93% advanced very high resolution radiometer (AVHRR) multi-channel image data and 7% of miscellaneous data. In compressing the image portion, we classify each pixel into 10 different groups and apply a multi-channel prediction and a non-linear error conversion. The entropy coder is an arithmetic coder which is adaptive and regenerates the approximation of the statistical properties of the source as an initial probability table. To compress the non-image part, we used the general compressor (gzip). From experimental results, the original information is compressed down to 25% $\sim$ 40%.

1 Introduction

The remotely sensed NOAA satellite "high resolution picture transmission" (HRPT) data provide very useful and important information in meteorology, oceanography and many other scientific fields. To date there have been many studies on image compression, particularly on lossy and very low bit rate compression. For image databases, a high compression ratio is important for storage and also for rapid transmission, but to deal with various kinds of users demands lossless image transmission is indispensable.

Also for this HRPT data, we must store them as they are. But one difficulty of this data is its size: one datum has more than 90MBytes, and we receive 5 $\sim$ 8 data each day. Reducing the size of the stored data is desired by both archiver and receiver.

The dominant part of this HRPT data is AVHRR (advanced very high resolution radiometer) image data. So we utilize the property of multi-channel 2-dimensional data of AVHRR data for compression. In the literature, some approaches of lossless compression (ex.[4, 7]) have been presented, but they are not very efficient in terms of compression ratio. For our database purposes, the compression ratio is more important than compression time, because decompression is a more common operation than compression.

In this paper we propose a method to losslessly compress the HRPT data which is somewhat computationally expensive but compresses much better.

<table>
<tr><td colspan="2">Header(ID,time,etc.)</td><td>1500 bytes</td><td rowspan="11">One line of HRPT data
22180 bytes</td></tr>
<tr><td>Channel 1</td><td>2 bytes</td><td rowspan="10">One line of
AVHRR data
20480 bytes</td></tr>
<tr><td>2</td><td>2 bytes</td></tr>
<tr><td>3</td><td>2 bytes</td></tr>
<tr><td>4</td><td>2 bytes</td></tr>
<tr><td>5</td><td>2 bytes</td></tr>
<tr><td>Channel 1</td><td>2 bytes</td></tr>
<tr><td>2</td><td>2 bytes</td></tr>
<tr><td colspan="2">⋮
repeat 2048 × 5 times</td></tr>
<tr><td colspan="2">Footer(synchronize)</td><td>200 bytes</td></tr>
</table>

Table 1: HRPT data format of one line

1.1 The Satellite NOAA and Its HRPT Data

The meteorological satellite NOAA-11 and NOAA-12 go around the earth at the average altitude of 810km in about 101.2 minutes. They have AVHRR sensor on board, which has five channels covering the wavelength from 0.55μm (channel 1, visible) to 12.5μm (channel 5, infra red). Reception of the observation data requires about 13 minutes when it is at its highest orbit and one transmission yields $3,000 \sim 4,000$ data lines. The word size for HRPT data is 10bits, but for simplicity of handling, our receiving system represents it by sixteen bits (two bytes). The 6 MSB's are padded with '0's. Therefore each HRPT data amounts to 90Mbytes. We receive $5 \sim 8$ HRPT data a day, so the total amount in a year exceeds 1TByte.

Each line of HRPT data is independent from all others, and contains 2048 pixels. The structure is shown in table 1.

Figure 1 shows an example of an AVHRR image obtained from NOAA-HRPT data.

2 Our Method

In this section our method will be explained. In short, our method for compressing AVHRR image portion is a kind of predictive coding such as DPCM. But we use many kinds of techniques to reduce its entropy and to code efficiently.

2.1 Noise-Line and Non-Imagery Part Treatment

Usually the AVHRR data contains noise lines(ex.figure 2). The number of noise lines in each datum is independent from all others. Such lines should be detected and removed from the image array, put together and compressed using a non-imagery compression method (gzip).

Figure 1: Example of AVHRR image (Channel 4, 2048 $\times$ 3736)

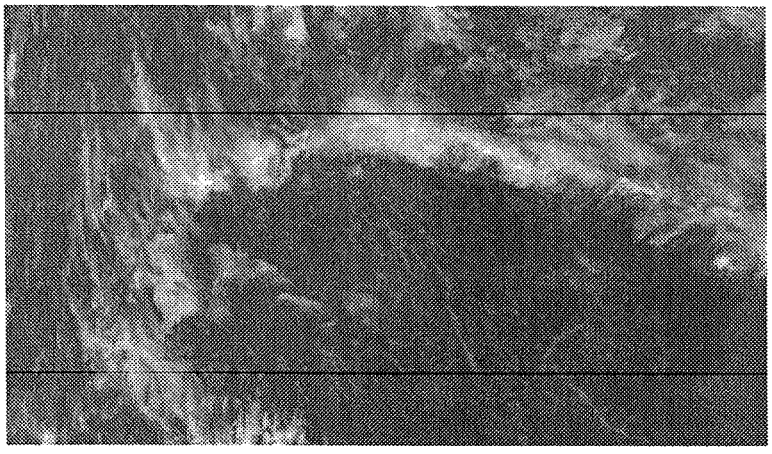


Figure 2: Example of noise lines in AVHRR data (A magnified part of channel 3, 544$\times$314). The two black horizontal lines are the noisy lines.

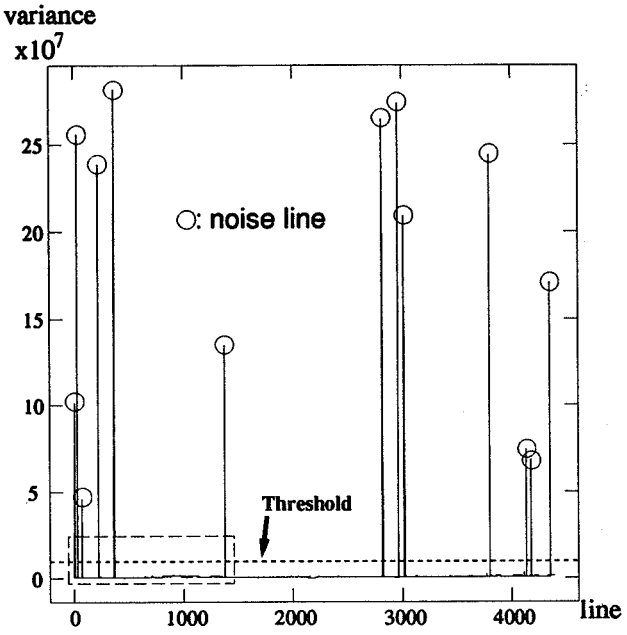


Figure 3: Calculated variance for each line. circles: detected noise lines, dotted line: threshold, rectangle with broken line: magnified area in figure 4

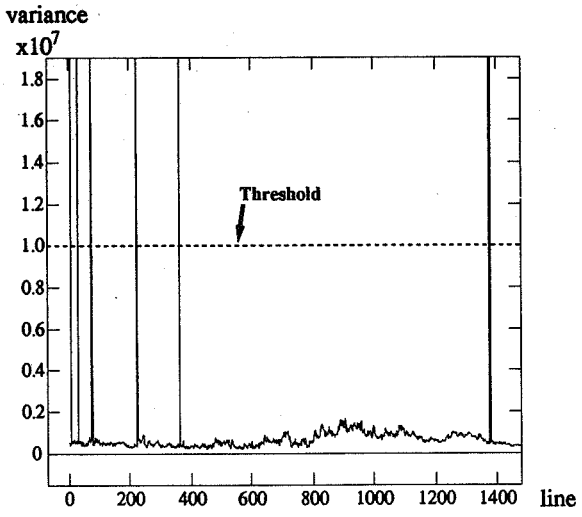


Figure 4: Magnified part of figure 3

To detect the noisy lines, we use a simple but reliable criteria. First we check the *ID* bytes in the header area. If the ID of a line is irregular, we regard the line as noisy. If a line passes this test, we calculate the variance of the difference between horizontally adjacent pixels in channel 1. If the variance is greater than a certain threshold (we use 1×10^7), we regard it as a noisy line. From experimental results, the first ID check is noise sensitive enough (see figure 3 and 4). We just use this second check to be doubly sure. The time for this check is significantly shorter than the total processing time.

The HRPT data contains about 7% of non-imagery data such as the fixed ID code, time stamp, fixed synchronizing code and so on (see table 1). The fixed or predictable portion is cut off. The remaining unpredictable portion and the noise lines are compressed separately from the image data. They are passed to gzip and compressed. Gzip is invoked with the '-9' option, which specifies the best compression.

This process is HRPT data dependent.

2.2 Pixel Classification

Each image pixel has different properties under certain criterions. From the point of image compression, grouping similar-propertied-pixels and encoding them respectively generates effective results. For grouping the pixels, we use the Q value:

$$Q \equiv |P_2 - P_1| + |P_3 - P_1| + |P_4 - P_1| + |P_5 - P_1| \quad (1)$$

Q	0–1	2–3	4–7	8–15	16–31	32–63	64–127	128–255	256–511	512–
Group #	0	1	2	3	4	5	6	7	8	9

Table 2: Grouping table

The position of the pixels ($P_1 ... P_5$) are shown in figure 5. This can also be calculated during decoding process, because only the upper or left pixels are used in equation 1. Using this Q value, we classify each pixel into several groups according to table 2.

2.3 Multi-Channel Prediction

For each classified group, we predict the value of the current pixel using linear combination of its neighbors' pixel values. The coefficients are calculated by the least square error method and use a constant to let the mean error be zero.

The neighbor pixels used for prediction are shown in figure 5 ($P_1 \ldots P_{10}$). For the pixels in the first channel, we use these 10 neighboring pixels.

The already decoded pixels are used to predict the pixels in the next channel ($R_1 \ldots R_9$ in figure 5). Using these pixels, something like interpolative prediction is achieved. This prediction contributes to the compression.

It is possible to use more than one of the previous channels if they are already decoded, but this increases processing time. We consider that looking for the optimal encoding order is more important. This is what we are investigating now.

2.4 Error Conversion

As each pixel has 10 bits, the prediction error $e(= \hat{x} - x)$ can have a real number value between -1024 and $+1024$ (roughly). After prediction, e should be expressed as an integer. An easy way to convert is to simply round the value off to an integer (calculate $\lfloor e + 0.5 \rfloor$) and consider it as a 2's complement 10-bit number.

Our conversion algorithm is somewhat different from simple rounding. After this

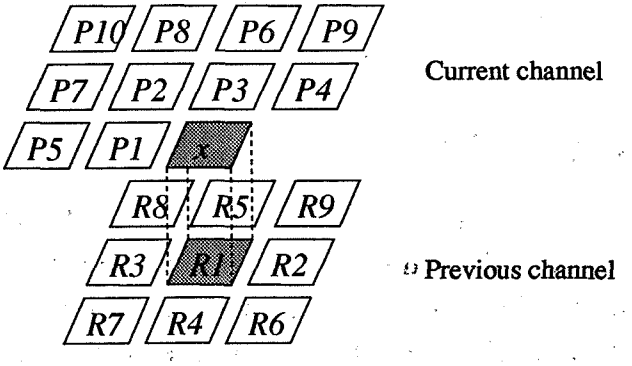


Figure 5: Pixels used for prediction

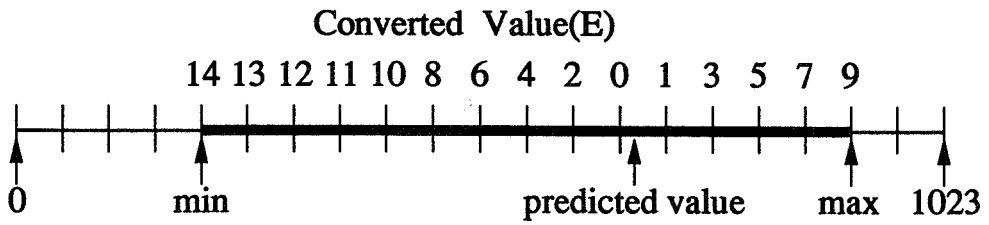


Figure 6: Algorithm for error conversion

conversion, we can also get the 10-bit non-negative integer E. First we obtain the upper and lower bound of the group (max, min). Then, as in figure 6, convert the prediction error into an positive integer. Within [min, max], the closest integer from the predicted value corresponds to $E = 0$, the second closest integer corresponds to $E = 1$, ... , and the n th closest integer corresponds to $E = n - 1$. (In figure 6, if the actual pixel value is equal to 'max', then $E = 9$.) For each group, we get the maximum and minimum pixel value and convert the prediction error respectively.

This conversion is reversible. If you get the predicted value and the converted number E (and also upper and lower bound), you can obtain the actual pixel value from the similar numerical rule.

2.5 Distribution Fitting and Entropy Coding

For natural images, the distribution of E is well approximated by the Gaussian distribution[8]. This distribution is used to generate the initial probability table for the encoder and decoder. The Gaussian distribution requires only one parameter – the variance – to be regenerated.

Figure 7 shows the graph of E vs. normalized distribution (probability) for each group (0...9). They do not exactly have the Gaussian shape, but for approximation and initial distribution generation, Gaussian curve fitting works well to reduce the code size. And it is clearly seen that from this figure, the curve of lower group (lower Q value) has more accurate peak (less variance) than that of upper group.

For the entropy coding, we adopt an arithmetic coder, because it has very effective performance and it is easy to make it adaptive.

3 Experimental Results

We programmed the compression program in C, on an HP9000/735. In this section the compression performance of our method and the time needed.

Table 3 lists the compression results made on 11 HRPT data obtained in December 1993. The column "stored size" means the whole file size of archived HRPT data in bytes, line-number $\times$ 22180 (the size of HRPT single line). The column "original size"

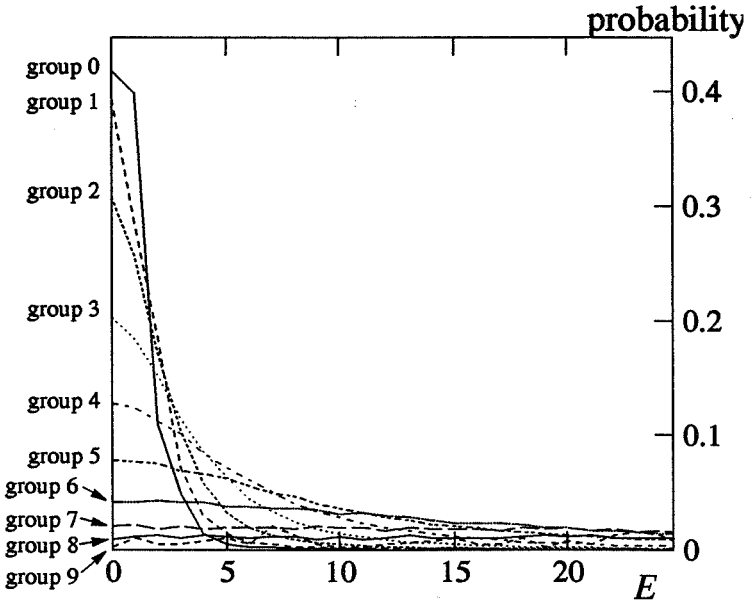


Figure 7: E vs. probability curve

means the actual amount of bits from NOAA satellite in bytes, "stored size" $\times \frac{10}{16}$. This difference is because we store one HRPT word (ten bits) by two bytes (see subsection 1.1). This value is used to calculate the column "compression ratio" (C.R.), i.e. "original size" divided by "compressed size".

In table 4 the comparison with 'gzip -9' is shown. Also the compression ratio is given by the ratio of original size to compressed size.

Date	lines	stored size	original size	C.R.	time(sec)
Dec.4, 15JST 1993	4400	97592000	60995000	3.219	4586
Dec.5, 15JST 1993	3390	75190200	46993875	3.506	3621
Dec.5, 16JST 1993	3023	67050140	41906337	3.458	3134
Dec.6, 15JST 1993	4400	97592000	60995000	3.219	4655
Dec.6, 16JST 1993	3607	80003260	50002037	3.245	3801
Dec.7, 14JST 1993	4289	95130020	59456262	3.277	4535
Dec.7, 16JST 1993	4023	89230140	55768837	3.309	4247
Dec.8, 14JST 1993	4358	96660440	60412775	3.338	4567
Dec.9, 14JST 1993	4087	90649660	56656037	3.194	4245
Dec.9, 16JST 1993	4400	97592000	60995000	3.189	4600
Dec.9, 17JST 1993	3053	67715540	42322212	4.713	3167

Table 3: Results of compression ratio (C.R.) and processing time

Date	lines	C.R.(gzip)	C.R.(proposed) and time(sec)
Apr.2, 15JST 1993	4400	1.204	3.018 (3303)
May 7, 14JST 1993	3187	1.188	3.066 (2381)
May 20, 15JST 1993	4400	1.137	2.730 (3297)
May 13, 20JST 1994	3267	1.686	4.280 (2522)

Table 4: Compression comparison with gzip and time

4 Conclusion

In this paper we proposed an effective method of lossless compression of NOAA HRPT images. Our method accomplishes the compression ratio of around 3 to 4. It actually means that in our receiving system the amount of HRPT data is reduced down to around one fifth. Though we don't use the same image as in other experiments found in the literature, the compression ratio by Kim's method[4] is around 2, and Tate's method for AVHRR data is around 2.7[7].

It is possible that the encoding order of channels has effect on compression ratio. Currently we encode five channels simply in channel order(1,2,3,...), and we only use the previous channel's pixels for prediction. Tate reports that for multi-spectral image, a well-chosen encoding order performs as well as optimal order[7]. He also reports that the effect of channel ordering makes slight difference especially for AVHRR data. We will seek for the optimal order and investigate if it is applicable to our data, and also examine using more channels for prediction.

It usually takes about one hour to compress a single HRPT datum. On the other hand, decompression is around six times faster. As mentioned in the introduction, the compression ratio matters more than the compression time, but this time might be considered too long. Therefore we are thinking of faster, more efficient and less redundant encoding algorithms.

References

[1] D. A. Huffman: "A method for the construction of minimum redundancy codes", Proceedings of IRE 40, pp. 411–420, 1951

[2] J. J. Rissanen et al.: "Arithmetic coding", IBM Journal of Research and Development, 23(2), pp. 188–193, 1976

[3] J. Ziv and A. Lempel: "A Universal Algorithm for Sequential Data Compression", IEEE Transactions on Information Theory, IT-23(3), pp. 337–343, May 1997

[4] M. Kim, et al.: "NOAA Data Compression Using a Multi-Length DPCM Code and a Variable-Length Code", Proc. 11th Asian Conf. Remote Sensing, P-1, 1990

[5] P. G. Howard and J. S. Vitter: "New Methods for Lossless Image Compression Using Arithmetic Coding", Proc. Data Compression Conference '91, pp. 257–266, 1991
[6] T. Taniguchi et al.: "Variable-Length-Code-Selective Reversible Predictive Coding for Multi-Level Images", The Transactions of the Institute of Electronics, Information and Communication Engineers, Vol.J70-B, pp.654–663, Jun. 1987
[7] S. R. Tate: "Band ordering in Lossless Compression of Multispectral Images", Proc. Data Compression Conference '94, pp. 311–320, 1994
[8] S. Takamura and M. Takagi: "Lossless Image Compression with Lossy Image using Adaptive Prediction and Arithmetic Coding", Proc. Data Compression Conference '94, pp. 166–174, 1994

The Development of Lossless Data Compression Technology for Remote Sensing Applications[1]

Pen-Shu Yeh and Warner H. Miller
Goddard Space Flight Center, Code 738.3
Greenbelt, MD 20771
T: 301.286.4477 F:301.286.1751 EMail:psyeh@psy.gsfc.nasa.gov
T:301.286.8183 F:301.286.1751 EMail:whmiller@gsfcmail.nasa.gov

Lossless data compression has been studied for many NASA missions to achieve the benefit of increased science return; reduced onboard memory requirement, station contact time and communication bandwidth. This paper first addresses the requirement for onboard applications and provides rational for the selection of the Rice algorithm among other available techniques. A top-level description of the Rice algorithm will be given, along with some new capabilities already implemented in both software and hardware VLSI forms. The paper then addresses systems issues important for onboard implementation including sensor calibration, error propagation and data packetization. The latter part of the paper provides several case study examples drawn from a broad spectrum of science instruments including the thematic mapper, x-ray telescope, gamma-ray spectrometer, acousto-optical spectrometer.

INTRODUCTION

With the development of new advanced instruments for remote sensing applications, sensor data will be generated at a rate that not only requires increased onboard processing, storage capability, but imposes demands on the communication link and ground data management system. Data compression provides a viable means to alleviate these demands. Two types of data compression have been studied by many researchers in the area of information theory: a lossless technique that guarantees full reconstruction of the data, and a lossy technique which generally gives higher data compaction ratio but incurs distortion in the reconstructed data. To satisfy the many science disciplines NASA supports, lossless data compression becomes the priority for technology development in this area.

To implement a data compression technique on the spacecraft, several criteria are considered:

1. The algorithm has to adapt to the changes in data to maximize performance.
2. It can be easily implemented with few processing steps, small memory and little power.
3. It can be easily interfaced with a packetized data system without performance degradation.

There exist a few well known lossless compression techniques including Huffman code, arithmetic code, Ziv-Lempel algorithm and variants of each. After extensive study and performance comparison on the same test image data set (Venbrux, 92)(Yeh, 91, 93), the Rice algorithm originated at Jet Propulsion Laboratories (Rice, 79) is selected for implementation.

1. Part of the paper is taken from NASA Technical Paper 3441, "Application Guide for Universal Source Encoding for Space," by the authors, Dec. 1993 and was presented in the International Geoscience and Remote Sensing Symposium, 94.

The Rice algorithm is essentially a set of Huffman codes organized in a structure that does not require lookup tables. The set of the Huffman codes can be easily extended to the information range of the science data. It is adaptive to the changes in the statistics of the data, and can be easily implemented. The structure of the algorithm also permits simple interface to data packetization scheme without having to carry side information across packet boundary. Therefore its performance is file size independent.

In 1991, a hardware engineering model was built in an Application Specific Integrated Circuit (ASIC) for proof of concept. This particular chip set was named as Universal Source Encoder/ Universal Source Decoder (USE/USD) (Venbrux, 92). Later, it was redesigned with several additional capabilities and implemented in Very Large Scale Integration (VLSI) circuits using gate arrays suitable for space missions. The flight circuit is referred to as Universal Source Encoder for Space (USES). The fabricated USES chip is capable of processing data up to 20 Msamples/second and will take data of quantization from 4-bit to 15-bit (MRC, 93).

A description of the Rice algorithm will be given in the next section, followed by systems issues and case study examples on remote sensing data either acquired from launched spacecrafts or simulated for future missions.

THE RICE ALGORITHM ARCHITECTURE

A block diagram of the architecture of the Rice algorithm (Rice, 91, 93) is given in Figure 1. It consists of a preprocessor to decorrelate data samples and subsequently map them into symbols suitable for the following stage of entropy coding. The entropy coding module is a collection of options operating in parallel over a large entropy range. The option yielding the least number of coding bits will be selected. This selection is performed over a block of J samples to achieve adaptability to scene statistics. An Identification (ID) bit pattern is used to identify the option selected for each block of J input data.

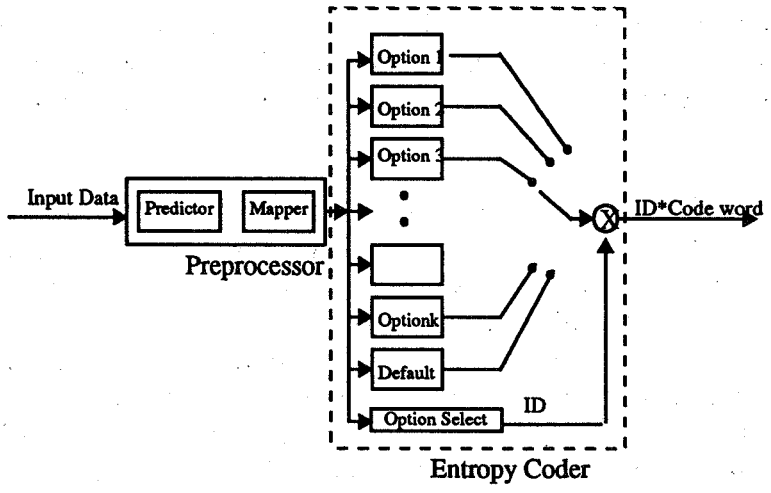


Figure 1. The encoder architecture.

The Preprocessor

The predictor in the preprocessor can be as simple as a first order predictor using previous sample, or other higher order predictors. To maintain the pipeline processing in the hardware, only a few predictor types are implemented, these include: a 1D predictor, a 2D predictor, a multispectral predictor and a user-supplied external predictor.

The function of the predictive coder is to decorrelate the incoming data stream by taking the difference between data symbols. The mapper takes these difference values, both positive and negative, and orders them, based on predictive values, sequentially into positive integers.

Entropy Coder

Most of the options in the entropy coder are called "sample-split options". These options take a block of J preprocessed data samples, split off the k least significant bits, and code the remaining higher order bits with a simple comma code before appending the split-off bits to the coded data stream. Each sample-split option in the Rice algorithm is optimal in an entropy range about one bit/sample (Yeh, 93); only the one yielding the least amount of coding bits will be chosen and identified for a J-sample data block by the option select logic. This assures that the block will be coded with one of the available Huffman codes, whose performance is better than other available options on the same block of data. The $k = 0$ option is optimal in the entropy range of 1.5 - 2.5 bit/sample; the $k = 1$ option is optimal in the range of 2.5 - 3.5 bit/sample, and so on for other k values.

To improve the performance below 1 bit/sample, a new option is devised and included in the full set of options implemented in VLSI. This new option is particularly efficient over data with very low entropy values.

The default option is an option not to use any of the split-sample options or the low-entropy option. It bounds the performance of the algorithm by simply passing through the preprocessed block of data through the encoder without alteration but with an appended identifier.

SYSTEMS ISSUES

Several systems issues related to embedding data compression scheme onboard a spacecraft should be addressed. These include the relation between the focal-plane array arrangement and the data sampling/prediction direction, the subsequent data packetizing scheme and how it relates to error propagation in case of bit error incurred in the communication channel and how the packetization may affect compression performance.

Sensor Calibration

Advanced imaging instruments and spectrometers often use arrays of individual detectors arranged in a 1D or 2D configuration; one example is the Charge Coupled Device (CCD). These individual detectors tend to have slight differences in response to the same input photon intensity. For instance, CCDs usually have a different gain and dark current value for each individual detector element. It is important to have the sensor well calibrated so that the data reflects the actual signals received by the sensor. Simulations have shown that for CCD types of sensors, gain and dark current variations as small as 0.2% of the full dynamic range can render the prediction

scheme less effective for data compression. Besides calibration, in order to maximize the compression gain, the 1D prediction scheme will be much more effective when data acquired on one detector element is used as the predictor for data acquired on the same detector whose characteristics are stationary, in general, over the data collection period.

Error Propagation

User acceptability to distortion resulting from channel bit errors is mission dependent, and is strongly a function of received Signal-to-Noise-Ratio (SNR), the data format, the error detection and correction technique and the percent of distorted data that is tolerable. A major concern in using data compression is the possibility of error propagation in the event of a single bit error in the compressed data stream. During the decompression process, a single bit error can lead to a reconstruction error over extended runs of data points. A general approach to minimizing this effect is to provide a very clean channel by using error detection/correction scheme. For compressed data stream, this still does not prevent error propagation, if it occurs, across a decompressed scanline. Further protection can be achieved by using packet data structure in conjunction with a properly chosen error correction scheme as advocated in the Consultative Committee on Space Data Systems (CCSDS) Blue Book (CCSDS, 89). Using this scheme, decompression error resulting from bit-error will then be contained in a packet for compression algorithms that do not carry side information across packet boundary.

Packetization and Compression Performance

Packetization is used not only as a means to contain bit error locally as just mentioned, it is a logical way to facilitate the transport of variable length bit string as a result of entropy coding. The Rice algorithm chooses an option for every block of samples, its performance is optimized within this block and there is no need to pass side information or statistics across packet boundary. There exist other compression techniques whose performance depends on establishing long-term statistics in a file. These schemes will give good compression performance for a large file and poor performance for a relatively smaller file. When packetization is used in conjunction with these algorithms to prevent error propagation, one would expect better compression performance for a larger packet. The drawback is that the loss of data caused by bit error may be intolerable.

CASE STUDY EXAMPLES

This section contains several compression study results for several different instruments. The compression performance is expressed as Compression Ratio (CR). It is defined as the ratio of the quantization level in bits to the average code word length, also in bits. It should be noted that the CR value is data dependent and can vary from one test data set to the next.

Landsat Thematic Mapper

Mission Purpose: The Landsat program was initiated for the study of Earth's surface and resources. Landsat-1, 2, and 3 were launched between 1972 and 1978. Landsat-4 was launched in 1982, and Landsat-5 in 1984.

Landsat Thematic Mapper (TM) on Landsat-4 and 5: The TM data represent typical land observation data. An image acquired on Landsat-4 at 30m ground resolution for band 1 in the wavelength region of 0.45 - 0.52 μm is shown in Figure 2. This 8-bit 512x512 image was taken over Sierra Nevada in California.

Compression Study: Using a 1D predictor in the horizontal direction, setting a block size of 16 samples and inserting one reference per every image line, the lossless compression gives a compression ratio at 1.83 for the 8-bit image.

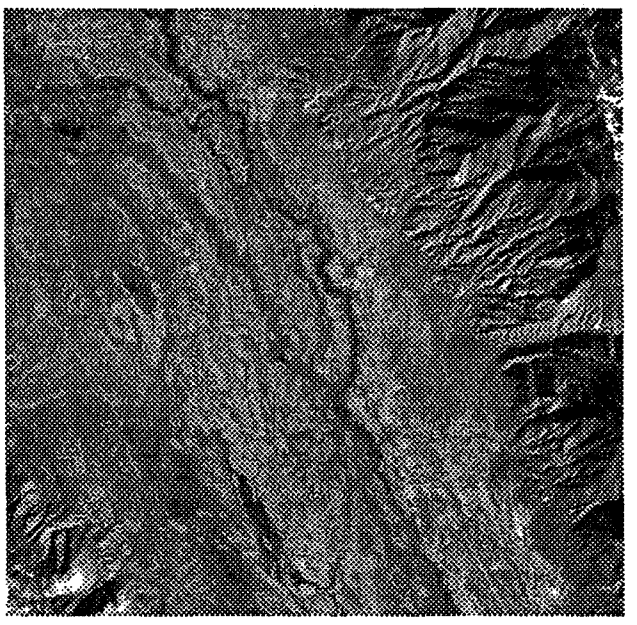


Figure 2. Thematic Mapper image

Soft X-ray Telescope (SXT) on Solar-A Mission

Mission Purpose: The Solar-A mission, renamed as Yohkoh mission after its successful launch in August, 1991, is dedicated to the study of solar flares, especially of high-energy phenomena observed in the X- and gamma-ray ranges.

Soft X-ray Telescope (SXT): The instrument detects X-ray in the wavelength range of 3-60 Angstrom. It uses a 1024x1024 CCD detector array to cover the whole Solar disk. Data acquired from the CCD is of 12-bit quantization and is processed on board to provide 8-bit telemetry data. The image in Figure 3 is an averaged image of size 512x512 with dynamic range up to 15 bits in floating point format as a result of further ground processing.

Compression Study: The test image is first rounded to the nearest integer. Then a 1D predictor is applied to this seemingly high-contrast image. A compression ratio of 4.69 is achieved.

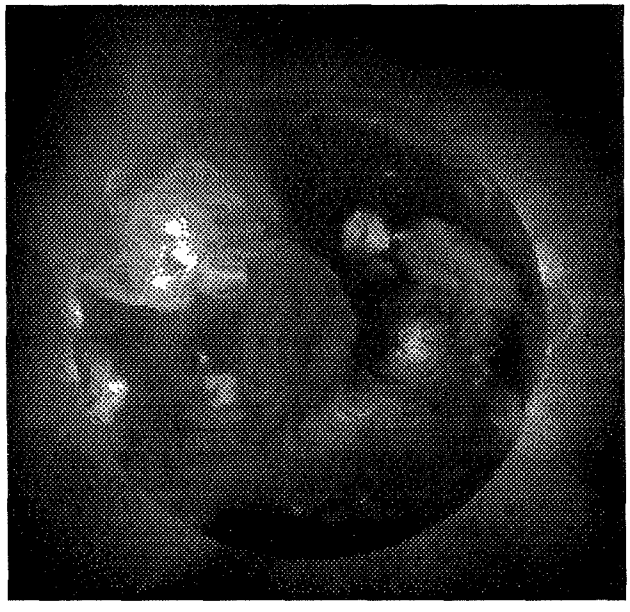


Figure 3. Solar-A X-ray image

Acousto-Optical Spectrometer (AOS) on Submillimeter Wave Astronomy Satellite (SWAS)

Mission Purpose: The Submillimeter Wave Astronomy Satellite (SWAS) is a Small Explorer (SMEX) mission, scheduled for launch in the summer of 1995 aboard a Pegasus launcher. The objective of the SWAS is to study the energy balance and physical conditions of the molecular clouds in the Galaxy by observing the radio-wave spectrum specific to certain molecules.

Acousto-Optical Spectrometer: The AOS utilizes a Bragg cell to convert the radio frequency energy from the SWAS submillimeter receiver into an acoustic wave, which then diffracts a laser beam onto a CCD array. The sensor has 1450 elements with 16-bit readout. A typical spectrum is shown in Figure 4(a). An expanded view of a portion of two spectral traces is given in Figure 4(b). Because of the detector nonuniformity, the difference in the Analog-to-Digital Converter (ADC) gain between even-odd channels, and effects caused by temperature variations, the spectra have nonuniform offset values between traces, in addition to the saw-tooth-shaped variation between samples within a trace. Because of limited available onboard memory, a compression ratio of over 2:1 is required for this mission

Compression Study: 1D prediction between samples is ineffective when the odd and even channels have different ADC gains. Using the multispectral predictor mode, the achievable CR is 2.32.

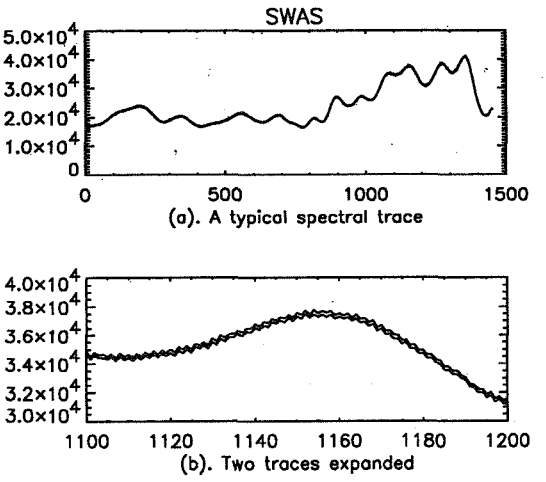


Figure 4. AOS radio wave spectrum

Gamma-Ray Spectrometer on Mars Observer

Mission Purpose: The Mars Observer was launched in September 1992. The Observer will collect data through several instruments to help the scientists understand the Martian surface, atmospheric properties and the interactions between the various elements involved. In the summer of 93, contact with the spacecraft was lost.

Gamma-Ray Spectrometer (GRS): The spectrometer uses a high-purity germanium detector for gamma rays. The flight spectrum is collected over sixteen thousand channels. The total energy range of a spectrum extends from 0.2 Mev to 10 Mev. Typical spectra for a 5-second and a 50-second collection time are given in Figure 5. These spectra show the random nature of the count. The spectral count dynamic range is 8-bit.

Compression Study: The achievable compression depends on the channel collection time. At 5-second collection time CR is over 20 and it decreases as collection increases. At 20-second collection time, CR is over 10.

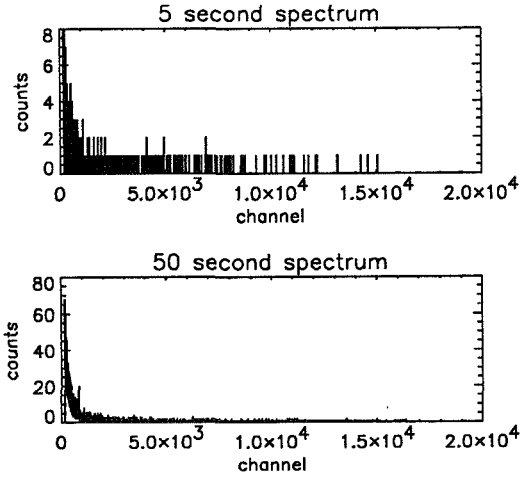


Figure 5. Gamma-ray spectrum

CONCLUSION

A lossless data compression technology has been successfully developed for remote sensing applications. This technology is based on the enhanced Rice algorithm. The performance of the algorithm has been established through analysis and simulation. Hardware in VLSI form as well as software are currently available for space flight missions. Over a dozen case studies have been performed on post-flight data and several new missions have adopted the technology for onboard implementation.

REFERENCES

CCSDS, "Advanced Orbiting Systems: Networks and Data Links: Architectural Specification," CCSDS 701.0-B-1 Blue Book, Oct. 1989. Available from CCSDS Secretariat, Communications and Data systems Division (Code-TS), National Aeronautics and Space Administration, Washington, DC 20546, USA.

MRC, "Universal Source Encoder for Space - USES," Preliminary Product Specification, Version 2.0, Microelectronics Research Center, University of New Mexico, 1993.

Rice, Róbert F., "Some practical universal noiseless coding techniques," *JPL Publication 79-22*, 1979. Available from author, JPL, 4800 Oak Grove Drive, Pasadena, CA. 91109, USA.

Rice, Robert F., Pen-Shu Yeh and Warner H. Miller, "Algorithms for a Very High Speed Universal Noiseless Coding Module," *JPL Publication 91-1*, 1991. A shortened version "Algorithms for High-Speed Universal Noiseless Coding," is published in the *Proceedings of the AIAA Computing in Aerospace 9 Conference,* San Diego, CA, Oct. 19-21, 1993

Venbrux, Jack, Pen-Shu Yeh and Muye N. Liu, "A VLSI Chip Set for High-Speed Lossless Data Compression," *IEEE Trans. on Circuits and Systems for Video Technology,* Vol. 2, No. 4, Dec. 1992.

Yeh, Pen-Shu, Robert F. Rice and Warner H. Miller, "On the Optimality of Code Options for a Universal Noiseless Coder," Revised version, *JPL Publication 91-2,* 1991. A shortened version "On the Optimality of a Universal Noiseless Coder," is published in the *Proceedings of the AIAA Computing in Aerospace 9 Conference*, San Diego, CA, Oct. 19-21, 1993.

Landsat Pathfinder Tropical Forest Information Management System

W. Salas*, W. Chomentowski*, J. Harville+, D. Skole*, K. Vellekamp*

*Complex Systems Research Center, University of New Hampshire, Durham, NH 03824
+Reseach Computing Center, University of New Hampshire, Durham, NH 03824

BACKGROUND

Tropical Deforestation is a *real world* problem that is scientifically significant and policy-relevant. In the last twenty years, the systematic destruction of tropical forests has become a global scale problem warranting attention from both scientists and policymakers. In terms of science it has been consistently singled out as a key element of many areas of global change research, including: global carbon cycle and climate change, biomass burning and atmospheric chemistry, and land surface water and energy balance. In terms of policy it is a central component of such high level initiatives as the Framework Convention on Climate Change, the Intergovernmental Panel on Climate Change, international tropical timber trade negotiations, and the General Agreement on Tariffs and Trade (the so-called GATT agreements).

The concern over tropical deforestation arises because of its potential influence on climate change and its general impact on the global environment. If deforestation continues at the current rate as much carbon dioxide and other greenhouse gases will be put into the atmosphere in the next 75 years as has been put into the atmosphere in the last 300 years and the potential for climate change will increase. Recent scientific findings suggest that deforestation can also influence climate change by altering sensible and latent heat flux, planetary albedo, and surface roughness at the planetary boundary layer. More local effects include increases in the fraction of precipitation as surface run-off, soil erosion, and an eventual local decline in precipitation.

Perhaps the greatest irreversible change associated with deforestation is the loss of biodiversity from habitat destruction and fragmentation. Some estimates suggest that the current rates of deforestation could result in the loss of up to one half of the world stock of genes, which would dramatically reduce the biological diversity of the plant and animal species and would severely limit the future of genetic stocks for biotechnology development.

Existing programs are obtaining the necessary earth science datasets. The Humid Tropical Forest Inventory Project (HTFIP) is the main component of NASA's Landsat Pathfinder Program. For two years it has been acquiring large amounts of high resolution Landsat data and has been mapping deforestation. When complete the project archive may be as much as 1,000 Gigabytes. This archive provides complete Landsat coverage with less than 20% cloud cover for tropics in South America, Central Africa, and Southeast Asia for three points in time: late 1970s,

mid 1980s, and early 1990s. The project has been acquiring data from the US national archives, foreign ground stations, and programmed acquisitions. Already the information produced by the project has made policy and scientific impacts.

However, to increase its usefulness, this information must be readily accessible. The raw data and derived products from HTFIP are important for scientists, policy makers, and educators. Because the HTFIP image library is large and stored at a single location, it is essential to provide tools that make browsing the library possible and make the library available over a high speed network. An Information Management System which incorporates digital library technology could make the information available on the Internet. Such development would ideally be targeted to three primary user communities: (a) earth scientists who need access to low and high level primary data usually in the form of satellite imagery, (b) policy makers who need access to the derived products and distilled information and relevant ancillary information usually in the form of digital maps, summary statistics, and published papers (and occasional sample images), and (c) educators and students (K-12) who need highly distilled or synthesized information more in the form of an on-line multi-media encyclopedia.

These themes echo those inherent in the National Information Infrastructure (NII) concept. We emphasized in our development approach that the Tropical Forest Information Management System (TFIMS) would make earth science data simultaneously relevant and accessible by a wide range of users, from young students to active scientists. We have had first hand experience in this regard through our involvement in developing the first test of the NII. Under the umbrella of the National Information Infrastructure Testbed the University of New Hampshire and Sprint collaboratively developed a prototype of the Landsat Pathfinder TFIMS last year.

INTRODUCTION

A Tropical Forest Information Management System has been designed to fulfill the needs of HTFIP in such a way that it tracks all aspects of the generation and analysis of the raw satellite data and the derived deforestation dataset. The system is broken down into four components: satellite image selection, processing, data management and archive management. However, as we began to think of how the TFIMS could also be used to make the data readily accessible to all user communities we realized that the initial system was too project oriented and could only be accessed locally. The new system needed development in the areas of data ingest and storage, while at the same time being implemented on a server environment with a network interface accessible via Internet. This paper summarizes the overall design of the existing prototype (version 0) information management system and then presents the design of the new system (version 1). The development of version 1 of the TFIMS is ongoing. There are no current plans for a gradual transition from version 0 to version 1 because the significant changes are in how the data within the HTFIP will be made accessible to the extended community of scientists, policy makers, educators, and students and not in the functionality of the basic system.

VERSION 0: EXISTING PROTOTYPE TFIMS

Version 0 has three distinct modules: query and browse, data management, and archive management. The query and browse section enables a user to locally search both US and foreign archive image metadata. The data management module is the project accounting system used to track imagery through the processing stream until it is archived. The archive management module picks up where the data manager leaves off by providing an interface to the data archive and a vehicle by which a user can explore the data.

Query and Browse: The query and browse module is a tool to graphically search TFIMS online metadata libraries. Two libraries are available for exploration, a large library containing global coverage from Landsat, SPOT, and the India Remote Sensing (IRS) satellite, or the smaller HTFIP library. The global library contains the metadata for all US Landsat holdings (approximately 790,000 MSS and 200,000 TM scenes), as well as holdings from all foreign Landsat ground stations that report to the Landsat Ground Station Operators Working Group (approximately an additional 700,000 MSS and 500,000 TM scenes). In addition, the library contains metadata for three Landsat receiving stations that have not reported to LGSOWG: Thailand, Ecuador and India. The global library also contains metadata for all IRS-1A and IRS-1B data and all the metadata for SPOT XS data acquired over the tropics. To our knowledge this is the most comprehensive metadata library for this type of imagery and is a valuable and important part of the TFIMS.

A single metadata entry contains 55 seperate items describing the image. The items provide information about the sensor, satellite, date of acquisition, identification, satellite reference system, geographical position of its center point, percent cloud cover, overall quality of the image, how the scene was recorded, etc. Some scenes will not have entries for all of its items due to differences in the sensors and ground station standards. A "no data" value is assigned to those items to insure that the user understands that information for that entry is not available. While these 55 items provide detailed information, there is no substitute for being able to visually inspect the image. Hence, availability of digital browse products would greatly enhance the usefulness of the metadata. There is a concerted effort in the Landsat community to create browse products for the historical archive and for all new acquisitions. Therefore, the HTFIP libraary will contain a browse product for each of its approximately 2700 Landsat images.

To search the metadata library with the query and browse tool, pull-down menus are used to define a query with constraints on geographic region, date, cloud cover and/or a number of other image descriptors. The query result is displayed as one or many rectangular polygons outlining the image footprint. Other data layers can be displayed simultaneously such as a regional coastline, vegetation, and towns. If a more detailed view of a selected scene is desired, a compressed picture, called a browse product, can be displayed by clicking on a footprint of interest. Figure 1. highlights both functions of this module by showing the result of a query for data availability from the archive at the EROS Data Center that are within Brazil for a specific date, image quality, and cloud cover. The geographic extent of all scenes that met the user defined

search criteria are displayed in red over the outline of South America. The two inserts are browse products of two scenes contained within the HTFIP archive.

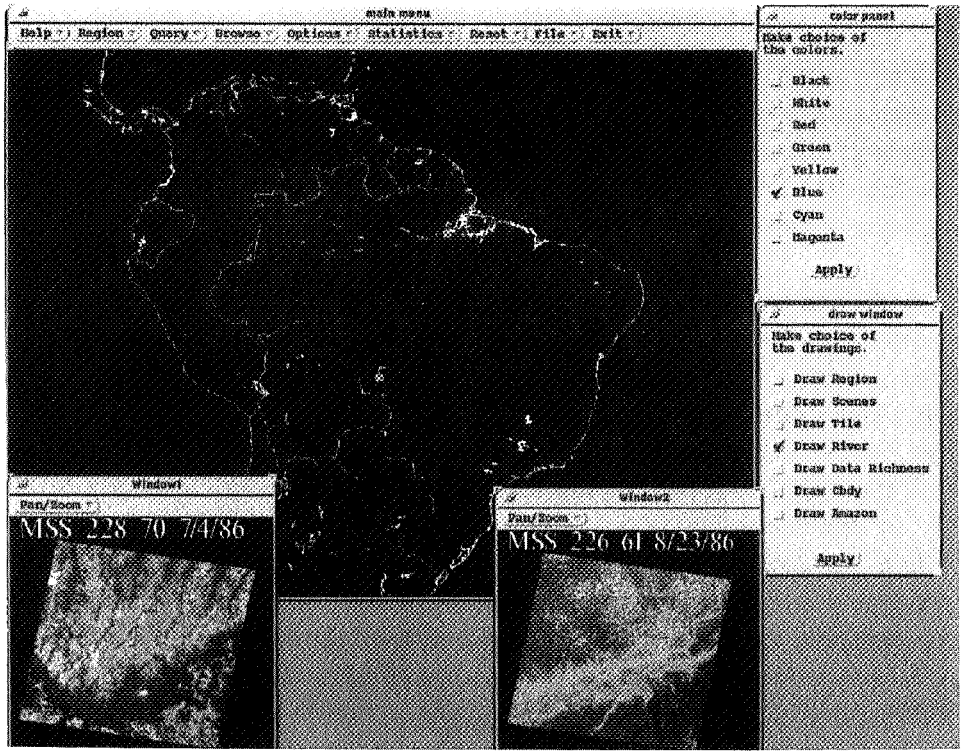


Figure 1. Query and Browse functions of the TFIMS.

Data management: The data management system (DMS) is similar to package tracking systems used by express mail companies but instead of tracking a package from origin to destination, the data manager tracks imagery through each phase of the processing stream and provides detailed information about individual scenes. Imagery is received by mail and entered into the data manager upon arrival. Each image is described by 144 attributes stored in a database management system (DBMS). The attributes include all the metadata items used to describe the scene such as acquisition date, path, row, as well as project specific information such as date ordered, date received, processing status, and map projection. The DBMS is internally linked to rectangular polygons in Arc/Info, representing the image boundary/footprint. The data manager can be queried to answer myriad questions, with the answers displayed graphically or in a tabular report. Questions may include whether the image has been ordered, the date ordered, date received, what phase of processing the image is in, as well as processing parameters, such as the clustering technique used to derive the deforestation map.

The end-to-end processing of individual scenes is broken up into five phases to facilitate its tracking. These phases are recorded in the TFIMS and are summarized as follows:

phase 0: scene has been ordered for HTFIP.
phase 1: scene has been received and passed quality control.
phase 2: scene has been digitally classified and converted from raster to vector.
phase 3: scene has been manually edited based on visual interpretation at 1:250,000 scale.
phase 4: scene has been georegistered and edgematched with its neighbors.

When an image has been ordered it is added to the database via the graphical user interface and considered to be in phase 0. Pertinent information is recorded which includes the aforementioned fields path, row, year, month, day, and sensor as well as other information such as region and data source. When an image arrives the inventory control specialist (ICS) updates the DBMS with quality control data and verifies that the image was ordered. Each image product arrives in a package which includes an 8mm tape containing the data descriptor record (DDR) and the digital data. The DDR is read directly from the tape to the DBMS and includes information such as the unique scene identification code, corner point coordinates in Universal Transverse Mercator (UTM) projection, and UTM zone. This information is initially used to match the new scene with the order request. Once the image passes this quality control step it is in phase 1 of the processing stream.

Upon completion of a phase, information necessary to reach that phase is entered by the ICS. For example, entering phase 2 information involves updating image processing parameters such as threshold values or clustering reclassification values as well as output histogram values and analyst name and date. In the future this information will be entered into a batch file which will be accessed weekly to update the DBMS automatically. Currently for phase 3 and phase 4 the date of completion is recorded. Further revisions will include information on initial and final numbers of polygons for each output class for phase 3 and move parameters for edge-matching for phase 4.

The user may query the DBMS for information regarding a particular image or for more information regarding the project inventory as a whole. The DMS is equipped to produce lists of scenes received, scenes sent to other processing centers, the processing phase of an image and scenes canceled due to inaccuracies in the metadata. Alternately, the user may enter the graphics mode to display this same information graphically utilizing the link to the scenes geographic information. The displayed image footprints may be overlaid on other geographically referenced information such as country boundaries, other satellite data, or vegetation maps. These displays can then be saved as postscript files for hardcopy outputs.

Archive Management: Managing the project archive effectively is an integral part of the data base. The archive will consist of almost 2700 Landsat MSS and TM scenes spanning a wide geographic area and a twenty year time period. In addition to imagery the archive will contain ancillary information such as ground truth data, scientific papers and allow access to wide area networks (WAN). The system to manage this archive consists of a hardware component to store the data and a software component to browse the archive. The storage system hardware combines three media types, magnetic disk, 8 mm tape and magneto optical. The system is able to store 500 Gb and provide near real time access. The storage system is linked to the network via

a data server. The software component is built around a commercial off the shelf (COTS) geographic information system. It provides an easy to use, graphical interface to the archive.

Before entering the archive management module it is assumed that the user has browsed the metadata library with the query and browse section and has chosen an image to examine closely. The archive manager does not have the capability to browse the whole library, it is used to explore one or more images in detail. Access to multi-media ground truth data or wide area networks is available through objects on the screen or pull down menus. Multi-media ground truth data which include photographs taken with a 35mm camera at the site, an audio description, and a written description are visualized by the archive manager.

Data recorded on site is linked to the georeferenced imagery by locations recorded in the field with a global positioning system (GPS). Upon invoking the archive manager the previously chosen satellite scene appears on the screen with data collection sites. The user can focus on an area of interest by zooming and panning around the image. To visualize data collected on the ground a point of interest is chosen with the mouse. Each location is internally linked to digitized photos, audio, and text. After clicking on the location all available ground data from the site of choice is displayed or in the case of audio transmitted through a built in speaker. Currently photographs are digitized with a scanner however photo cd technology is being implemented for use by the archive manager. Currently, the archive manager contains data collected by scientists at UNH. Links to detailed data bases outside of UNH at organizations such as The Nature Conservancy and The Missouri Botanical Gardens are being developed.

Within the archive manager a user can access WAN tools such as Mosaic and Gopher. Such a capability enables access to national library card catalogs and on line data from most scientific research centers. Mosaic and Gopher are started with a pull down menu. The archive manager has a small internal library containing scientific journal articles on subjects pertinent to research at UNH. We are developing a collaborative browse capability using a high speed WAN so that scientists at remote sites can analyze a data set simultaneously. With a collaborative browse tool two or more scientists view the same data set simultaneously discuss it, overlay other data sets, and communicate over an audio and video link.

The archive manager is used to store multi media data, to access the HTFIP data library, and to visualize satellite and ground data. It is being used operationally in the Landsat Pathfinder project to assist in photo interpretation. It can also be used by scientists working on global change or students interested in the tropical forests. It is a more effective way to store and visualize multi-media data than slides in a three ring binder with written notes and locations. It can also be a useful scientific tool because two scientists, thousands of miles apart will be able to visualize the same data simultaneously.

Figure 2. is an example of how the GPS locations are displayed on the imagery with the corresponding photographs, field notes, and link to the WAN. This figure depicts several of the key functions of the archive management system. On the left side are several GPS points depicted as green boxes with cross hairs and overlayed on Landsat MSS imagery. One of the points has been selected (shown in white) and the field notes are displayed in the text window with two

slides taken at the point displayed below the text window. On the right side of the figure are the links to the WAN via Gopher and Mosaic.

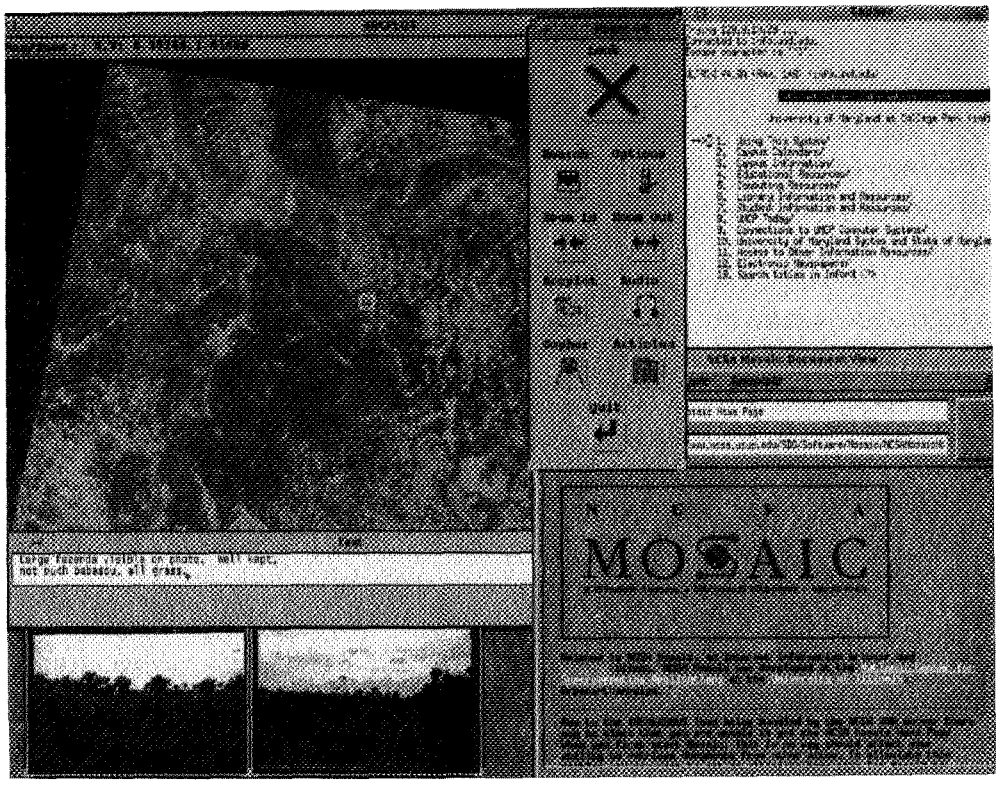


Figure 2. Archive Management functions of the TFIMS

LANDSAT PATHFINDER TFIMS: VERSION 1

Why redesign the existing prototype information management system? While the prototype described above met the initial needs of our tropical deforestation mapping project, a more elegant and efficient system is being designed to enable the system to be accessed and used by a diverse group of users. To facilitate this we plan to make the new system faster and accessible over the Internet. The redesign of the prototype focuses on two broad areas: implementation of more sophisticated data ingest and storage techniques and development of the system environment. This section of the paper describes the planned development for the new system.

Data Ingest and Storage: To efficiently utilize large databases of satellite imagery and associated derived products, sophisticated data ingestion and compression techniques are required. In addition, to make the data truly accessible and usable for the many various users the data must be

made available in a variety of formats. While development of compression techniques is developing rapidly, sufficient capabilities exist now to handle these databases in an elegant and timely manner.

We plan that version 1 will utilize both lossy and lossless compression techniques. The need for both types of compression can be seen in the following two examples. For browse products generation, the benefit from higher compression ratios associated with lossy techniques will more than offset the degraded image quality of the reconstructed browse images. However, some visualization capabilities will require reconstruction of full resolution lossless images. For example, the images from the digital library for depicting location of *in situ* ground data need to be accurately reconstructed at full resolution to enhance integration of the different data layers.

Lossy compression techniques include JPEG and Sarnoff methods. JPEG uses a predictive modeling technique based on differential pulse code modulation with varying, user defined, compression quality settings. Higher compression ratios can be achieved using lower quality settings. Success of predictive modeling techniques is dependent on the degree of correlation within the dataset, therefore, the high spectral and spatial correlation within satellite datasets bids well for these techniques. We plan to test all eight predictors available within JPEG to assess which predictor(s) tend to work well with Landsat imagery from the tropics.

Lossless compression techniques will be required to display, at user defined resolutions, *in situ* ground data and other spatial datasets simultaneously with satellite data. The basic theory behind lossless compression is to remove all redundancy (or correlation) within the dataset and is accomplished in two phases: decorrelation and coding. Several decorrelation techniques will be evaluated with each type of dataset in this system to design the most efficient models. These techniques include dictionary based modeling (like the Lempel-Ziv algorithm used by the UNIX compress command) and predictive modeling (differential pulse code modulation with various predictors). We plan to evaluate Huffman and Arithmetic coding based on their speed in reconstructing the imagery.

Another important capability of the lossless compression techniques to be examined is the efficiency (speed) at which compressed full resolution images can be reconstructed at various user defined resolutions. This need for multiresolution display capabilities arises from the wide variation in the spatial scale of analyses and datasets. We plan to explore how efficiently various decorrelation and coding methods work within the context of multiresolution display.

We expect the success of the decorrelation and coding techniques to vary significantly due to distinct approaches among the algorithms and the inherent differences in the datasets. However, the format of the datasets and the data ingest and retrieval techniques will also influence the speed of the compression, decompression, and the compression ratios. Since Hierarchical Data Format, or HDF, is the current choice for the storage format for EOS-DIS, it is imperative that these techniques are evaluated on HDF data sets. For example, images are stored Science Data Sets within HDF, and, therefore, are stored as band sequential (BSQ) files. The compression ratios for images stored as BSQ will be different than if the same image were stored as a band interleaved file due to differences in correlation between pixels adjacent in spectral or spatial space. Our

evaluation of the various existing compression techniques will drive what format the datasets will be stored in. In an effort to provide the datasets in a suite of output formats we are developing a series of translators to provide the user with some flexibility in formats.

Version 1 System Environment: The Pathfinder TFIMS Version 1 will be accessible over the Internet and an Asynchronous Transfer Mode (ATM) wide area network (WAN). The system environment is composed of four main parts: data server; compute server; application server and network environment with connections to the Internet or an ATM WAN (Figure 3). System development will emphasize four components: data server, compute server, application server, and the network environment.

Data Server: The data server environment provides the device management and data storage functions of the system. This subsystem controls the file server and physical device access to the data archive. The data server environment includes: a UNIX server, magnetic disks, an optical disk storage device, an 8mm tape storage device and the compression algorithms involved in archiving data. The data server provides archive storage and access to the following categories of data and information: metadata, Landsat digital imagery, imagery analyses and synthesis data, field data, publications, supporting documents and a variety of multimedia information; and ancillary data and maps.

The magnetic disks provide a front end to nearly one terabyte of archived data/information on magneto optical disks and 8mm tapes. Users request archived data/information from specially configured file systems on the magnetic disks. Requests for archived data/information that are not currently present on the magnetic disks are delivered automatically, using robotic technology, to the magnetic disks from either the magneto optical disks or 8mm tapes. This is referred to as "near line" data. At this point the data/information remains directly accessible on the magnetic disks until a configuration parameter has been reached causing the data/information to be removed from the magnetic disks. Typically, this happens when the data/information has not been used for a defined period of time. When required, the data/information may be delivered directly to a locally attached disk on the compute server, application server or users workstation.

The actual requests for archived data and information are embedded in the TFIMS and are thus transparent to the user. The TFIMS presents a menu driven point and click graphical user interface (GUI) for users to select areas and types of data/information. The TFIMS converts these menu selections into requests for specific data/information and then sends requests to the data server. This approach lends itself well to a distributed computing environment (DCE) as there can be multiple data servers in different locations providing data and information seamlessly to the user. These details are hidden from the user and thus, provide a simple integrated access to the data and information for all users.

Compute Server: The compute server consists of four CPUs providing required data processing and I/O services. This server is used to manipulate and process metadata, raw image data, derived products, conduct analyses of collected data and for the development of multimedia data.

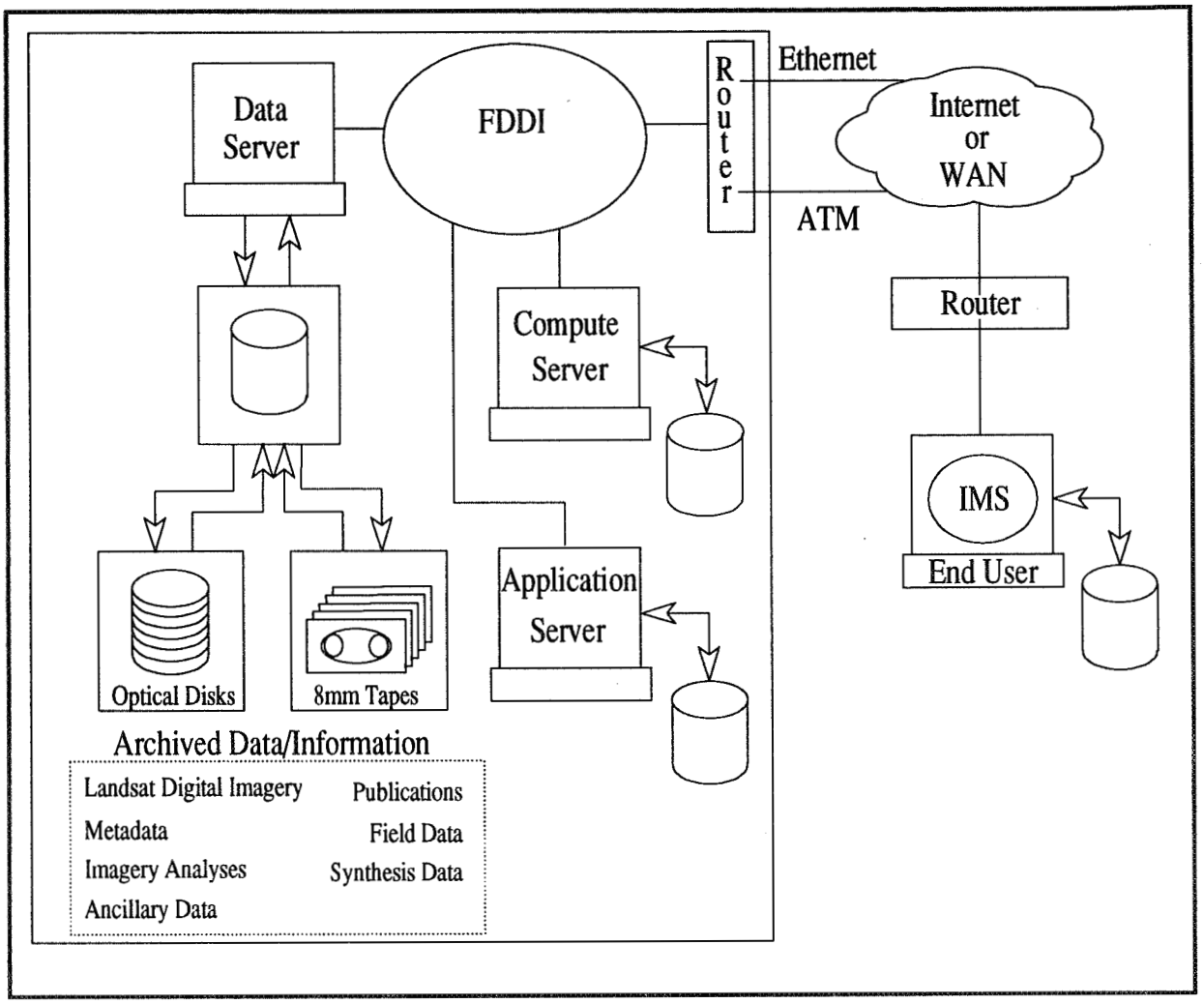


Figure 3. System Environment for the TFIMS

Application Server: The application server provides users with a GUI to the TFIMS, interfaces with the data server and compute server and handles user requests for ordering products. Version 0 relies on a licensed software product, ArcInfo. As the prototype develops into IMS Version 1, a Mosaic interface will be introduced to allow offsite user access via the Internet. Additionally, the reliance on ArcInfo will be minimized with the Mosaic version. Graphic images produced by ArcInfo will be saved in a format (e.g. GIF, TIFF, JPEG, HDF) compatible with common or publicly available graphics tools (e.g. xv). This will allow all image display data to be accessible to Mosaic users without the use of ArcInfo. The Mosaic interface will provide Internet users the ability to conduct query and browse operations of both metadata and imagery, order imagery and derived products and obtain ancillary multimedia information. IMS Version 1 (non Mosaic version) supported by ArcInfo will still be used internally to develop and track new products.

Network Environment: The network environment consists of local networks and network protocols interconnected via the Internet or an ATM WAN. Locally the network protocols and environment consist of an FDDI ring connecting all local servers, developer stations and on site end user stations. Additionally, the FDDI ring will be connected to a router which will provide remote users network access to the local network. The use of an FDDI ring locally provides transport of data at rates up to 100 MBs a second. This is ten times the transfer rate of ethernet thus, allowing for rapid timely transfer of large amounts of image data. For external connections to the local network, both an Internet and ATM connection will be available. The ATM connection will provide remote users with data transfer rates ranging from 45 MBs to 155 MBs per second.

The network interface module will be provided by using the Mosaic interface. Mosaic will allow applications and data to be distributed over the network on different servers at different locations all transparent to the user. As a model for a Science Computing Facility IMS, the TFIMS Mosaic version will allow the seamless integration of new functions and data from different sources/locations without burdening the user with knowing where the data and applications are and how to access them.

Planning/Scheduling Techniques for VQ-Based Image Compression

Nicholas M. Short, Jr.
Code 935, Goddard Space Flight Center, Greenbelt, MD 20771

Mareboyana Manohar
Hughes STX, Code 935
Goddard Space Flight Center
Greenbelt, MD 20771

James C. Tilton
Code 935, Goddard Space Flight Center, Greenbelt, MD 20771

Abstract

The enormous size of the data holdings and the complexity of the information system resulting from the EOS system pose several challenges to computer scientists, one of which is data archival and dissemination. More than ninety percent of the data holdings of NASA is in the form of images which will be accessed by users across the computer networks. Accessing the image data in its full resolution creates data traffic problems. Image browsing using a lossy compression reduces this data traffic, as well as storage by factor of 30-40. Of the several image compression techniques, VQ is most appropriate for this application since the decompression of the VQ compressed images is a table lookup process which makes minimal additional demands on the user's computational resources. Lossy compression of image data needs expert level knowledge in general and is not straightforward to use. This is especially true in the case of VQ. It involves the selection of appropriate codebooks for a given data set and vector dimensions for each compression ratio, etc. A planning and scheduling system is described for using the VQ compression technique in the data access and ingest of raw satellite data.

1 Introduction

Over the next decade, the rate at which data is generated by space-borne instruments will increase dramatically over current levels. A major contributor to this increase is the Earth

Observation System (EOS), planned for the end of this decade. The five proposed instruments on the EOS AM-1 platform and the six proposed instruments on the EOS PM-1 platform will generate data at a combined rate of 281 Gigabytes per day.

This raw data generated by the EOS platforms will be in turn processed into data products, including radiometrically and geometrically corrected images and a large number of science data products. This increases the data volume that must be handled and stored from the EOS instruments by an order of magnitude. Thus, over one Terabyte of EOS data products will be stored each day, along with other Earth science data, in distributed active archive centers (DAACs) located throughout the United States. Over the 15-year life of EOS, the archives will manage 11 petabytes of raw, processed, and analyzed data.

Success of future Earth science missions depends upon increasing the availability of data to the scientific community who will be interpreting space-based observations for issues such as ozone depletion and greenhouse effects, land vegetation and ocean productivity, and desert/vegetation patterns to name a few. Part of NASA's role in the Mission to Planet Earth (MTPE) initiative is to take a proactive leadership role in the management of space and Earth science data and in making those data accessible to scientists worldwide in order to foster the new field of Earth Systems Science.

Even at current data volumes, it is difficult to design and operate effective data archive and distribution systems for NASA Earth science data archives. With the increasing volumes of data that will be stored in these data archives, efficient browsing and distribution of data from these archives becomes even more important. An effective data archive and distribution system must give quick access to image browse and other data so users may quickly select the data required for their application. The availability of image data at intermediate resolution levels would also help users resolve ambiguities in the data selection process.

From our research in the Information Science and Technology Branch (ISTB), we present here an image browsing scheme using VQ and progressive VQ compression algorithms that we claim are excellent candidates for image data browsing and retrieval. A key feature of VQ and progressive VQ is their asymmetry in encoding and decoding. The minimal computational requirements of progressive VQ for decoding make possible very quick retrieval on moderate computer systems. The more computationally intensive encoding process can be accomplished, at a sufficient rate to keep up with the incoming data flow, in centralized data processing centers using more powerful computers, such as the recent massively parallel models.

To compress image data an expert level of knowledge is required. For example, a VQ or progressive VQ based image compression needs information about the data and the instrument the data belongs to, vector dimensions, etc. for selecting the codebook for compression. Usually the user has no knowledge of this information. However, the user is primarily con-

cerned about the compression ratio and quality of the compressed image. Therefore, a planning/scheduling system is required that accepts the user specified parameters and translates them to VQ related parameters. Thus the Planner/Scheduler essentially helps eliminate an image compression specialist from data dissemination process.

2 Image Compression

Image compression is one of many tools that can be used to help address Mission to Planet Earth's data handling challenges [23]. However, no single data compression approach is likely to be appropriate for all aspects of the problem. Lossless compression is required for data archiving, while some degree of information loss may be allowable for video image transmission. For image browse applications, larger amounts of information loss may actually be desirable. For browse, a general overall impression of the data quality and content may be all that is necessary, and a large reduction of data volume may be required. The key task for lossy data compression for browse applications is to preserve only the information required. Data characteristics also must be considered in designing an appropriate data compression approach, since data compression approaches often assume a particular data model.

Earth scientists often need to browse data to check the appropriateness and quality of particular data sets for detailed analysis. Further, appropriately derived browse data can facilitate interdisciplinary surveys which search for evidence of unusual events in several data sets from one or more sensors. In addition, browse data can be used to validate the quality of the data by facilitating quick checks for data anomalies. These different uses of browse data put possibly conflicting requirements on the browse data, and may require that separate browse data sets be produced for each major use category.

If a "progressive" data compression approach [23] is used, browse data can also facilitate the distribution of the data from the archive. Here the image is compressed at various levels called a compression hierarchy. The first level of the hierarchy provides an initial rendition appropriate for browsing the data. The ensuing levels of the hierarchy contain the details that are missing at earlier levels. Either a user or the planner/scheduler would inspect the browse data, and decide at "anytime" whether or not to inspect the data more closely. If a closer inspection is desired, additional levels of the compression hierarchies would be requested, until the user decides that data is not appropriate for the application and terminates accessing the data set, or until fully reconstructed data is obtained. Under this scheme, the data distribution process is kept efficient since no redundant information is ever sent or used.

Many image compression approaches show promise for the data archive and distribution problem. These include the Joint Photographic Experts Group (JPEG) standard lossless

and lossy compression methods [18], the Rice algorithm [19, 20], variations on Vector Quantization [10, 1, 14]. In addition, combinations of subband/wavelet decomposition and Vector Quantization [2, 3, 11], and combinations of subband/wavelet decomposition with the Karhunen-Loeve transform [8, 15] also show promise.

We have concentrated our efforts on investigating image compression based Vector Quantization. These approaches are particularly suitable for data archives and distribution across computer network applications due to asymmetrical coding and decoding efficiencies. The coding is computationally expensive, but is a one time effort, and can be performed at an archival center using a large capacity machine. The decoding part, however, is a computationally inexpensive table lookup process which does not burden the end user with computational difficulties.

2.1 VQ and Progressive VQ

VQ is the vector extension of scalar quantization which is found to be very useful for multispectral image compression ([13, 15]). The VQ vectors are obtained from image data by systematically extracting nonoverlapping blocks (typically 4x4) and arranging the pixels in each block in raster scan order. Such vectors allow VQ to exploit two dimensional correlations in the image data. If the image is multispectral, nonoverlapping cubes (typically 4x2x3) may be used. VQ builds up a dictionary of a few representative vectors, called codevectors, and then codes the image with the index value of the closest codevector from the dictionary, called codebook, in place of of each vector. Each codevector is represented by an address containing $log_2 M$ bits, where M is number of codevectors in the codebook. Assume vectors of size k are drawn from the input image and matched with those in the codebook. Using the indices of the matched codevectors to represent the input image vectors results in a decreased rate of $(log_2 M)/k$ bits/pixel or a compression ration of $(k * n)/log_2 M$, where n is the radiometric resolution of the image. In all practical situations the codebook size, M, is much smaller than the number of vectors that make up the input image.

The most important phase of VQ is the training process in which an optimal codebook (by some criterion such as least MSE) is learned from the input samples. The most widely used algorithm is Linde-Buzo-Gray (LBG) algorithm ([10]). Both the training and coding phases of VQ require finding the codevector which is closest match to a given vector. Computing this closest match requires computations proportional to the size of the codebook. Computational cost can be reduced by employing a suboptimal approaches such as Tree Search Vector Quantization (TVSVQ) and Pruned Tree VQ (PTVQ) [10]. The computational problems can also be solved by using a special architectures [13].

Progressive VQ [14] is a progressive variant of VQ in which multiple compression levels are provided. The first level is a VQ coding in which the codebook and codevector parameters are adjusted to give a relatively high compression ratio (e. g., in the range of 30 to 50). The image reconstructed from this first level coding can serve as browse data for a data archive system. If n levels are used, the second through the n-1 levels are VQ coded residuals. The nth level residual is not VQ coded, but instead is encoded with a lossless approach, such as the Rice algorithm [20] or Ziv-Lempel algorithm [25].

3 Planning and Scheduling for Image Compression

Given that image compression, like many other image processing routines, has many possible variants and uses, selection and coordination of the appropriate routines for particular users needs requires the use of a supervisor function. Many researchers have suggested the application of rule-based expert systems for capturing user requirements and knowledge for image processing[16, 17, 6, 21]. However, none of these techniques explicitly takes into account the computational complexity or the resource requirements for image processing tasks. In this domain where computational resources are constrained and hard deadlines for data acquisition exist, a better model that combines knowledge representation with resource modeling needs to be incorporated.

Recently, researchers have suggested the use of AI planning /scheduling techniques to manage the coordination of image processing operators such as image compression[7, 22, 5, 12]. For this paper, we will illustrate a particular planning /scheduling approach, called PlaSTiC, which is being used at the ISTB.

PlaSTiC was developed by the ISTB and Honeywell Technology Center as a planning /scheduling tool for a distributed computing environment. PlaSTiC is a hierarchical planner loosely based upon work by [24]. The core system is based upon the Honeywell's Time Map Manager (TMM) that handles reasoning about temporal information[4]. PlaSTiC combines the Nonlin planner[9], TMM, and extensions that allow for reasoning about the duration and resource requirements of plans[5].

For the image processing, plans are handed to an execution monitor which interprets plans according to the run-time environment, assigns uncommitted tasks to processes, and collects statistics for the planner. These statistics provide best-case/worst-case estimate intervals for primitive tasks and are propagated back up a task formalism [5] to provide better constraints during task decomposition.

As with most planners, PlaSTiC maintains a knowledge-base of plan operators that during planning, provides the necessary knowledge for plan construction. As an example,

contrived plan operator for PVQ compression, consider the following:

```
(opschema pvq-compression :todo (file-format ?FileID PVQ-COMPRESSED)
                          :expansion ((step1 :goal (file-format ?FileID BINARY))
                                      (step2 :goal (file-format ?FileID BSQ))
                                      (step3 :primitive (PVQ-COMPRESS ?name ?name ?cname
                                                        ?c ?r ?x ?y ?n IDM))
                                      (step4 :primitive (UNIX-COMPRESS ?name ?cname UNIX)))
                          :orderings ((before step1 step2) (before step2 step3) (before step3 step4))
                          :conditions ((:use-when (name ?FileID ?name))
                          (:use-when (size ?FileID (?r ?c)))
                          (:use-when (codebook ?FileID ?codebook))
                          (:use-when (codebook-name ?codebook ?cname))
                          (:use-when (vectx ?codebook ?x))
                          (:use-when (vecty ?codebook ?y))
                          (:use-when (codebook-band-number ?codebook ?n)))
:duration                 (range-addition (file-format-estimator 2)
                                          (pvq-estimator ?n ?r ?c ?x ?y))
:variables (?x ?y ?n ?r ?c ?FileID ?name ?cname ?codebook))
```

Essentially, the above pvq-compression operator states that in order to put a file (represented by the variable ?FileID) into PVQ compressed format (i.e., via the todo slot), two goals (i.e., step1 and step2) for putting the file in binary and binary sequential format must be done before the pvq-compress command (step3) gets called. In this case, each of the steps are totally ordered[1]according to the orderings slot. This operator is only applicable if there exists the appropriate information specified by the conditions slot.

In PlaSTiC, information about the duration of these operators is specified either explicitly through the duration slot above or through a statistical gathering mechanism that sets the duration of primitive steps (e.g., steps 3 and 4 above). Durational information is specified as a range of values from a lower bound to an upper bound. For operators with the duration slot, a function can be specified that must return a range. This function's arguments are derived through variables that are bound from the conditions slot[2].

Typically, the function in the duration slot is either a statistical estimator or a polynomial (e.g., *big oh* notation). Examples of the former can be as simple as returning the min/max of a working set or as complicated as output from an unsupervised clustering where attributes can be any property from the execution environment such as CPU utilization, machine type, input size, etc. For the primitive steps, durations are only min/max values from a working set.

[1](before step1 step2) means step1 occurs before step2

[2]Actually, unbound variables can exist as well, but that requires a more complicated mechanism

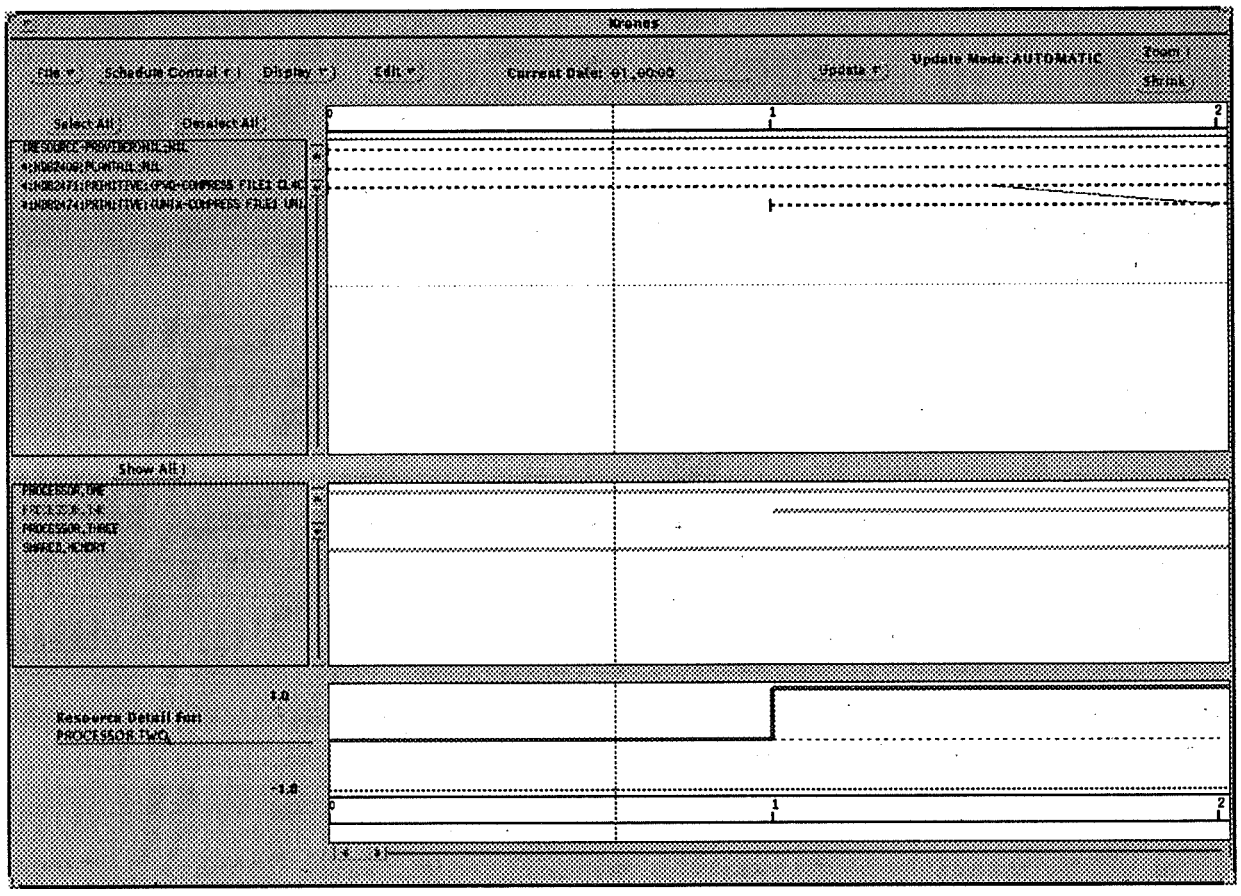


Figure 1: Plastic Output of Image Compression Plan

3.1 Planning for Image Compression

The current implementation of the image compression knowledge in the planner involves selection of the VQ or standard compression algorithms. If the compression technique selected is VQ, knowledge includes codebook selection, vector dimensions and host machine where the compression is executed. In particular, the compression knowledge is incorporated into a general image processing knowledge base for remote sensing data.

Specifically, when the image compression goal is a subgoal of another plan for data archiving, the planner chooses VQ codebooks and vector dimensions based upon user constraints on compression ratios and quality of compressed data. Figure 1 shows an example output from a very simple plan using the operators in the previous section. The interface shows potential resource subscription problems in the bottom two windows, while task intervals for the two steps and the orderings between them are shown in the top window.

We are currently addressing the problem of relaxing user constraints to fit the real-time constraints of ingest. In this case, the planner will continue to relax the compression parameters until both deadlines and resource constraints can be satisfied. To do this, a planning method of interleaving planning and execution will have to be incorporated into the ingest process. For example, progressive VQ requires the application of a particular quality level for the first level of compression to determine the next level's compression ratio. Selection of the codebook at each level must be initiated by the planner as a function of the previous algorithm application.

4 Conclusion

For a first pass, we have shown that Progressive VQ compression can be easily incorporated into the planning process. Because of the time and resource constrained environment of satellite processing, the choice of not only Progressive VQ compression techniques, but also other more traditional approaches, requires the use the coordination between a planner and a scheduler such as PlaSTiC. However, future systems that incorporate an interleaved planning/scheduling approach whereby results are checked during the planning processes are required for the Progressive VQ techniques.

Acknowledgments

The authors would like to thank Mark Boddy (Honeywell TC), Jim White (Honeywell TC), John Beane (Honeywell TC) for their invaluable contributions throughout the project.

References

[1] B.D. Andrews, P.A. Chou, M. Effros, and R.M. Gray. A mean-removed variation of weighted universal vector quantization for image coding. In *Proc. Data Compression Conference, IEEEE Catalog 93TH0536-3*, pages 302 - 309, Snowbird, UT, March 1993.

[2] M. Antonini, M. Barlaud, P Mathieu, and I. Daubechies. Image coding using wavelet transform. *IEEE Trans. on Image Processing*, 1(2):205 - 220, 1992.

[3] C. F. Barnes and E. J. Holder. Classified variable rate residual vector quantization applied to image subband coding. In *Proc. Data Compression (IEEE Catalog Number 93TH0536-3)*, pages 272 - 281, Snowbird, UT, March 1993.

[4] M. Boddy. Temporal reasoning for planning and scheduling. *Sigart Bulletin*, 4(3), July 1993.

[5] M. Boddy, J. White, R. Goldman, and N. Short, Jr. Planning applications in image analysis. In *1994 NASA-GSFC Proceedings of Space Applications of AI*, Greenbelt, MD, May 1994.

[6] D. Civco. Knowledge-based land use and land cover mapping. In *Proceedings ASPRS/ACSM*, volume 3, pages 276 - 91, 1989.

[7] R. F. Cromp, W. J. Campbell, and N. M. Short, Jr. An intelligent information fusion system for handling the archiving and querying of terabyte–sized spatial databases. In *International Space Year Conference on Earth and Space Science Information Systems.* American Institute of Physics, 1992.

[8] B. R. Epstein, R. Hingorani, J. M. Shapiro, and M. Czigler. Multispectral klt-wavelet data compression for landsat thematic mapper images. In *Proc. Data Compression (IEEE Catalog Number 92TH0436-6)*, pages 200 - 208, Snowbird, UT, March 1992.

[9] S. Ghosh, J. Hendler, S. Kambhampati, and B. Kettler. Common lisp implementation of nonlin user manual. Technical report, University of Maryland, Collage Park, MD, February 1991.

[10] R. M. Gray. Vector quantization. In *IEEE ASSP Magazine*, pages 4 - 28, April 1984.

[11] F. Kossentini, W. C. Chung, and M. J. T. Smith. Low bit rate coding of earth science images. In *Proc. Data Compression - IEEE Catalog Number 93TH0536-6*, pages 371 - 380, Snowbird, UT, March 1993.

[12] A. Lansky. A data analysis assistant. In *Proceedings of the 1994 AAAI Spring Symposium on Software Agents*, Stanford, CA, 1994.

[13] M. Manohar and J. C. Tilton. Progressive vector quantization of multispectral image data using a massively parallel simd machine. In *Proc. Data Compression (IEEE Catalog Number 92TH0436-6)*, pages 181 - 190, Snowbird, UT, March 1992.

[14] M. Manohar and J. C. Tilton. Progressive vector quantization on a massively parallel simd machine with application to multispectral image data. submitted to IEEE Transactions on Image Processing, 1994.

[15] T. Markas and J. Reif. Multispectral image compression algorithms. In *Proc. Data Compression - IEEE Catalog Number 93TH0536-6*, pages 391 - 400, Snowbird, UT, March 1993.

[16] D. Mckeown, W. Harvey, Jr., and J. McDermott. Rule-based interpretation of aerial imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(5):570 - 85, September 1985.

[17] B. Nicolin and R. Gabler. A knowledge-based system for the analysis of aerial images. *IEEE Transactions on Geoscience and Remote Sensing*, GE-25(3), May 1987.

[18] Pennebaker and Mitchell. *JPEG: Still Image Data Compression Standard.* Van Nostrand Reinholt, New York, 1993.

[19] R. F. Rice and J. R. Plaunt. Adaptive variable length coding for efficient compression of spacecraft television data. In *IEEE Trans. on Communication Technology*, pages 889 – 897, 1971. COM-19(I).

[20] R. F. Rice, P. S. Yeh, and W. H. Miller. Algorithms for a very high speed universal noiseless coding module. Publication 91–1, Jet Propulsion Laboratory, Pasadena, California, February 15 1991.

[21] N. Short, Jr. A real-time expert system and neural network for the classification of remotely sensed data. In *Technical Papers, 1991 ACSM-ASPRS Annual Convention*, volume 3. ASCM/ASPRS, 1991.

[22] N. M. Short, Jr. and L. Dickens. Automatic generation of products for terabyte-size geographic information systems using planning and scheduling. *accepted for publication in the International Journal of Geographic Information Systems*, 1994.

[23] J. C. Tilton, M. Manohar, and J. A. Newcomer. Earth science data compression issuess and activities. In *Remote Sensing Reviews*, 1994. 8.

[24] S. Vere. Planning in time: Windows and durations for activities and goals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(3):246 – 267, 1983.

[25] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Tran. Information Theory*, IT-23:337–343, 1977.

AN OVERVIEW OF THE EOSDIS V0 INFORMATION MANAGEMENT SYSTEM: LESSONS LEARNED FROM THE IMPLEMENTATION OF A DISTRIBUTED DATA SYSTEM

Patrick M. Ryan
Hughes STX Corporation
Greenbelt, MD
Phone: (301) 441-4261
E-Mail: ryan@odouls.stx.com

ABSTRACT

The EOSDIS Version 0 system, released in July, 1994, is a working prototype of a distributed data system. One of the purposes of the V0 project is to take several existing data systems and coordinate them into one system while maintaining the independent nature of the original systems. The project is a learning experience and the lessons are being passed on to the architects of the system which will distribute the data received from the planned EOS satellites. In the V0 system, the data resides on heterogeneous systems across the globe but users are presented with a single, integrated interface. This interface allow users to query the participating data centers based on a wide set of criteria. Because this system is a prototype, we used many novel approaches in trying to connect a diverse group of users with the huge amount of available data. Some of these methods worked and others did not. Now that V0 has been released to the public, we can look back at the design and implementation of the system and also consider some possible future directions for the next generation of EOSDIS.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1994	Conference Publication

4. TITLE AND SUBTITLE

Science Information Management and Data Compression Workshop

5. FUNDING NUMBERS

930

6. AUTHOR(S)

James C. Tilton, Editor

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Goddard Space Flight Center
Greenbelt, Maryland 20771

8. PERFORMING ORGANIZATION REPORT NUMBER

94B00116

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, D.C. 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

CP-3277

11. SUPPLEMENTARY NOTES

This workshop was organized and sponsored by the National Aeronautics and Space Administration (NASA) in Cooperation with the Institute of Electrical and Electronics Engineers (IEEE) Geoscience and Remote Sensing Society.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified–Unlimited
Subject Category 59
Report is available from the NASA Center for AeroSpace Information, 800 Elkridge Landing Road, Linthicum Heights, MD 21090; (301) 621-0390.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This document is the proceedings from the "Science Information Management and Data Compression Workshop," which was held on September 26-27, 1994, at the NASA Goddard Space Flight Center, Greenbelt, Maryland. The Workshop explored promising computational approaches for handling the collection, ingestion, archival and retrieval of large quantities of data in future Earth and space science missions. It consisted of eleven presentations covering a range of information management and data compression approaches that are being or have been integrated into actual or prototypical Earth or space science data information systems, or that hold promise for such an application. The workshop was organized by James C. Tilton and Robert F. Cromp of the NASA Goddard Space Flight Center.

14. SUBJECT TERMS

Data Compression, Image Compression, Information Management, Space Science, Earth Science

15. NUMBER OF PAGES

109

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

National Aeronautics and
Space Administration
Code JTT
Washington, D.C.
20546-0001

Official Business
Penalty for Private Use, $300

SPECIAL FOURTH-CLASS RATE
POSTAGE & FEES PAID
NASA
PERMIT No. G27

POSTMASTER: If Undeliverable (Section 158, Postal Manual) Do Not Return