NASA Conference Publication 3183

Space and Earth Science Data Compression Workshop

Proceedings of a workshop held at the Snowbird Conference Center Snowbird, Utah March 27, 1992

NASA

NASA Conference Publication 3183

Space and Earth Science Data Compression Workshop

James C. Tilton, *Editor*
NASA Goddard Space Flight Center
Greenbelt, Maryland

Proceedings of a workshop sponsored by the
National Aeronautics and Space Administration
and the IEEE Computer Society Technical Committee
on Computer Communications and held at
the Snowbird Conference Center
Snowbird, Utah
March 27, 1992

NASA
National Aeronautics and
Space Administration
Office of Management
Scientific and Technical
Information Program
1993

FOREWORD

The second annual Space and Earth Science Data Compression Workshop was held on March 27, 1992 in Snowbird, Utah. This NASA Conference Publication serves as the proceedings for the workshop. The workshop was held in conjunction with the 1992 Data Compression Conference (DCC'92), which was held at the same location March 24-26, 1992. The third annual Space and Earth Science Data Compression Workshop is planned for April 2, 1993, also at Snowbird, Utah.

The goal of the Space and Earth Science Data Compression Workshop series is to explore the opportunities for data compression to enhance the collection and analysis of space and Earth science data. Of particular interest is research that is integrated into, or has the potential to be integrated into, a particular space and/or Earth science data information system. Participants are encouraged to take into account the scientist's data requirements, and the constraints imposed by the data collection, transmission, distribution and archival system.

Papers were selected from submissions to the 1992 Data Compression Conference (DCC '92). Eleven papers were presented in 4 sessions. Discussion was encouraged by scheduling ample time for each paper, and through scheduled discussion periods at the end of each session.

The workshop was organized by James C. Tilton of the NASA Goddard Space Flight Center (GSFC), and Sam Dolinar of the Jet Propulsion Laboratory (JPL). Contact information is given below.

NASA Data Sets Available for Testing Data Compression Algorithms

The Space and Earth Science Data Compression Workshop held last year at the end of DCC '91 made several specific recommendations to encourage and enable the use of data compression technology in NASA missions. One recommendation was that "NASA should provide test data sets and examples of analysis scenarios to the data compression research community. These data sets should cover a broad range of NASA applications, concentrating on high data volume cases, and cases requiring high transmission bandwidth between the sensors and Earth, and across communications networks on Earth."

Just prior to the 1992 workshop, several test data sets were made available over "anonymous ftp" via Internet. In the near future, certain NASA data sets will be available on CD-ROMs. Contact the workshop organizers for information on how to obtain these CD-ROMs. Below is a discussion of the test data sets that were identified at the time of the workshop.

Large Scale AVIRIS and Landsat TM Data Sets

Jeff Dozier, the Earth Observing System (EOS) Project Scientist, made available two large NASA image data sets via "anonymous ftp" during March and April 1992: a Landsat Thematic Mapper (TM) data set and an Airborne Visible and Infrared Imaging Spectrometer (AVIRIS) data set.

The TM data (7 bands) are provided both as separate images for each band and as a band-interleaved-by-pixel image. The AVIRIS image (224 bands) is provided only as a band-interleaved-by-pixel image. They have been compressed with the UNIX 'compress' program

(Lempel-Ziv). The files are large: uncompressed, the AVIRIS image is 70.4 MB and the TM images are 12.6 MB in each band.

For information on these data sets, including how to obtain them, contact Jeff Dozier by electronic mail at dozier@crseo.ucsb.edu on Internet. If you wish to discuss testing of science algorithms on reconstructed images contact Walter Rosenthal (walter@crseo.ucsb.edu) about the TM data and Anne Nolin (nolin@crseo.ucsb.edu) about AVIRIS.

Collection of Small Test Data Sets

Warner Miller and Pen-Shu Yeh of the Goddard Space Flight Center have made available a variety of small test data sets that they have been using for data compression tests. While these data sets are small, they are representative of a wide range of NASA image data. Nine single-band test images are all of size 128-by-128. "clover.dat," "mount.dat," and "urban.dat" are 8-bit data with 3 meter ground resolution (clover => interstate cloverleaf, mount => mountains, and urban => urban area). This data was digitized from film taken by an airborne camera. "bay.dat", "bay_ir.dat," and "clouds.dat" are 8-bit data with 1 kilometer ground resolution (bay => visible band image of Chesapeake Bay, bay_ir => infrared band image of Chesapeake Bay, and clouds => image containing clouds). This data was obtained by the Heat Capacity Mapping Mission (HCMM) satellite in Application Explorer Mission 8 (AEM-8) in the late 1970's. "avi_cps1.dat," "avi_mfs1.dat," and "avi_mfs2.dat" are 12-bit scenes at 20 meter ground resolution from the August 31, 1988 flight of AVIRIS (mfs1 and mfs2 => subscenes 1 and 2 from the Moffet Field, and cps1 => a subscene from Cuprite, Nevada data set). Also included in this collection is a full scene (512 lines by 614 columns) of band 20 from the Cuprite data set, and a near full scene (312 lines by 614 columns) of band 20 from the Moffet Field data set. These data sets may be obtained through an anonymous ftp to chrpalg.gsfc.nasa.gov in subdirectory pub/miller. (Several of these images are reproduced as Figure 7 on page 91 of this document.)

Microgravity Experiment Data Sets

Daniel Glover at the NASA Lewis Research Center has offered to provide a collection of image data sets from Microgravity experiments. The data being considered will be digitized from 16mm color film (e.g., microgravity flame and droplet combustion experiments), and from color VHS video tape (e.g., from Shuttle fluids experiment). Contact workshop organizers for information on how to obtain this collection of test data sets.

Other Miscellaneous Data Sets

James Tilton at the NASA Goddard Space Flight Center has collected various NASA image data sets for use in his image analysis and data compression studies. These data sets may be obtained through an anonymous ftp to chrpalg.gsfc.nasa.gov in subdirectory pub/images. The data sets include 32 bands of a 210-band AVIRIS data set (bands 4 through 35, 512 lines by 614 columns), a Coastal Zone Color Scanner image (5-bands, 486 lines by 1536 columns), and several Landsat Thematic Mapper (TM) data sets of sizes ranging from 256 by 256 to 512 by 512. One group of Landsat TM data sets has corresponding ground reference files. The README file in the pub/images directory gives a more detailed description of the data sets and ground reference files.

Future Plans

Providing appropriate test data sets only partially fulfills the 1991 workshop recommendation; the provision of analysis scenarios was also recommended. It remains to be seen how NASA can effectively provide to data compression specialists guidance in analysis scenarios for the image data. Providing such guidance may not be in many cases where analysis scenarios are not well established. Nevertheless, it is hoped that by studying the data sets provided, data compression specialists will gain understanding of NASA's data compression needs, and report promising results at next year's workshop.

Acknowledgments

The organization of this workshop was supported by the Office of Aeronautics and Space Technology, NASA Headquarters, Washington, DC. Thanks also to the Office of Space Science and Applications for assistance in compiling the NASA data compression test data sets.

Workshop Organizers

James C. Tilton
Mail Code 930.1
NASA GSFC
Greenbelt, MD 20771
phone: (301) 286-9510
FAX: (301) 286-3221
Internet:
tilton@chrpisis.gsfc.nasa.gov
GSFCMAIL: JTILTON.

Sam Dolinar
Mail Stop 238-420
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109
phone: (818) 354-7403
FAX: (818) 354-6825
Internet:
sam@rtop71.jpl.nasa.gov

CONTENTS

PAPERS

APPENDIX

PAPERS

SUBBAND CODING FOR IMAGE DATA ARCHIVING

Daniel Glover
NASA Lewis Research Center
Cleveland, Ohio 44135

S. C. Kwatra
The University of Toledo
Toledo, Ohio 43606

Abstract. The use of subband coding on image data is discussed. An overview of subband coding is given. Advantages of subbanding for browsing and progressive resolution are presented. Implementations for lossless and lossy coding are discussed. Algorithm considerations and simple implementations of subband systems are given.

1. Introduction

Subband coding is a promising strategy for data compression that has been used successfully in recent years. This paper discusses subband coding with the perspective of organizing data for storage and retrieval rather than as just a data compression strategy. An overview of subband coding is given followed by discussions of subbanding for archiving. Finally, some simple algorithms for subbanding image data are shown.

Subband coding of speech was developed in the late 1970's [1] and matured in the 1980's. The key development for subbanding signals was the Quadrature Mirror Filter (QMF) in 1976 [2]. A QMF bank cancels out the aliasing that is generated by filtering providing a good quality reconstruction of the signal. Subband coding was applied to image data in the mid-1980's [3,4,5]. The theory for perfect reconstruction filter banks (which not only cancel aliasing, but do not introduce amplitude or phase distortion) was also developed around this time [6]. An overview of concepts important to subband coding can be found in [7,8,9].

2. Subband Coding

A generic subband coding system is shown in Figure 1. The original signal is split up into N frequency bands (subbands) by the analysis filter bank represented by the matrix $H(z)$. Each subband is then decimated, keeping only every Nth value. If the proper conditions are met, the resulting total number of values in the subbands is the same as in the original signal. No compression has been achieved yet; the subbanding process prepares the data for compression coding. The coders shown in each of the subbands could be the same or (more commonly) tailored to match the characteristics of the data in a particular type of subband.

After the data has been coded, it is transmitted through the channel and received or, equivalently, stored and recalled. The data is decoded and the subbands sent through a synthesis filter bank, $F(z)$, where the original signal is reconstructed. The synthesis filter bank has a special relationship with the analysis bank such that the aliasing caused by the analysis filters is canceled. The filter banks can be designed such that the reconstructed signal is a perfect replica of the original signal, provided that no errors have been introduced in the coding or transmission stages.

To take advantage of the two-dimensional structure of image data, the filters used in the subband system should be two dimensional. Designing such filters is difficult, so separable filters have typically been used to accomplish 2-D processing. Separable filters use 1-D filters and apply them to the data in two directions, i.e., first horizontally and then vertically. This results in

straightforward filter design, but requires storage to access horizontally processed data in a vertical direction (see Figure 2).

An important property of image data is the importance of phase as a vehicle for information. A good example is obtained from reconstructions of images using only the Fourier transform phase compared with reconstructions using only the magnitude; the phase-only reconstructions are recognizable whereas the magnitude-only reconstructions are not [10]. Linear phase filters are most commonly used for image data. Since most of the information is in the phase, frequency resolving capability is not as critical an issue with image data as it is with, say, audio data. This means that short kernel filters can be used on image data with good results.

The advantages of using short kernel filters include simple hardware design and speed of algorithm execution. Short kernel implementations are attractive for on-line compression and for fast retrieval of stored data.

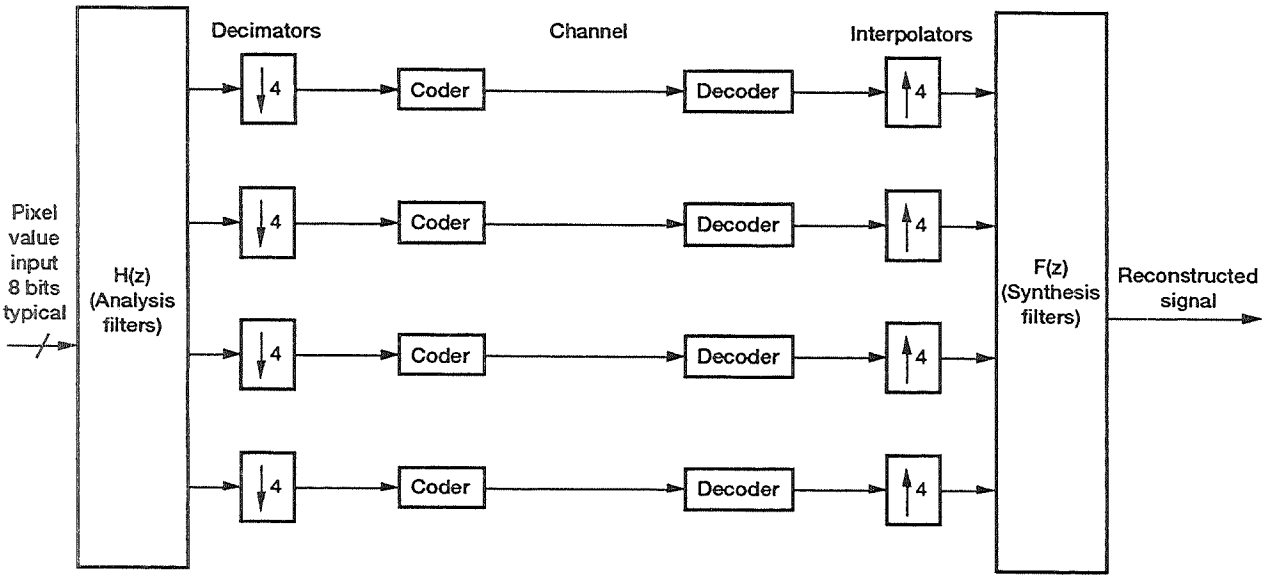


Figure 1. A generic four-channel subband coder.

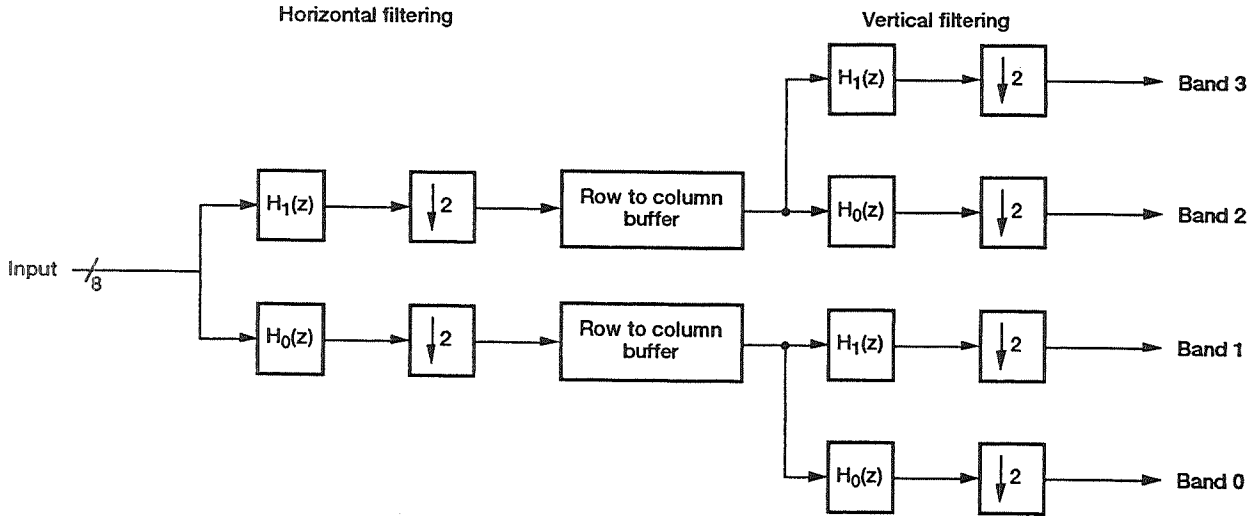


Figure 2. A separable, four-channel subband analysis bank.

3. Browsing and Progressive Resolution

A property of the subband signals that makes the splitting of the original signal into lowpass, bandpass, and highpass bands useful for lossy compression is that most of the original signal energy winds up in the low frequency band. The higher frequency subbands can be compressed quite a bit without compromising the picture quality if the lowest subband is not degraded.

The fact that the low frequency band is a lower resolution version of the original image makes subbanding attractive as a browsing tool. The low frequency band is generated as a consequence of subbanding and can be tracked separately from the higher frequency subbands in a browsing catalog. The low band can be retrieved and viewed separately. If a higher resolution version is desired, the higher frequency bands can be retrieved and image reconstructed.

Color images are handled by subbanding each component separately. These components could be RGB, YUV, etc. For browsing, the most significant component can be used (e.g., luminance for color images) or all the components can be retrieved and viewed separately or recombined into a low resolution color image. Care must be taken when subbanding chrominance data or false colors can be generated in the low resolution (low frequency) band.

Color image data can be stored as color components (RGB) or as luminance and chrominance data (e.g., YUV). The components are usually subbanded and coded separately. A subband filter that averages pixels for the low frequency band can generate false colors around transitions. To avoid this it is possible to use a low frequency subband that is just a subsampled version of the original (see Section 5).

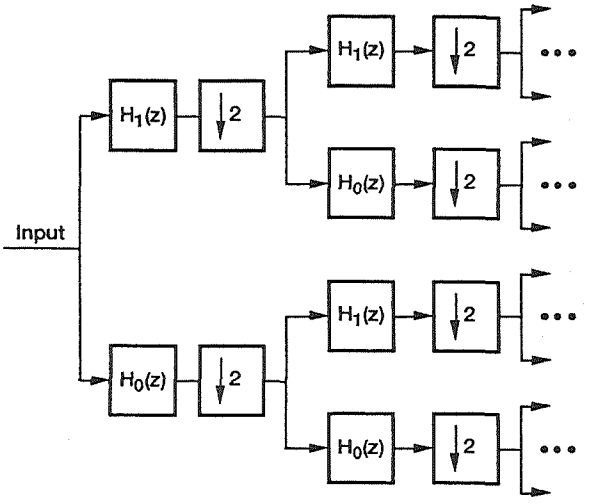
A typical approach to generating multiple subbands is to cascade sets of two-channel filters in a tree structure. Although it is possible to design multiple-band filter banks, it is usually easier to just cascade a simple two-band filter bank to get four, seven, sixteen, or whatever number of bands is appropriate. A corresponding tree structure is used for the synthesis bank and this lends itself to progressive resolution reconstruction.

The two most common tree structures are the octave-band split and the uniform-band split (see Figure 3). Either structure is equivalent from a browsing standpoint since the low frequency bands can be made as small as is practical for viewing using either tree. For progressive resolution, the octave-band split is most efficient in terms of reconstructing a series of increasing resolution images.

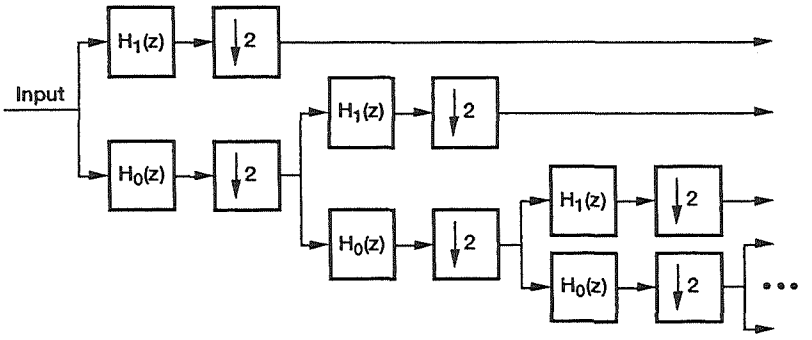
4. Lossless and Lossy Coding

Subbanding is a strategy for coding and as such can be used in either a lossless or lossy manner. The subbanding process itself can be made lossless by using perfect reconstruction filter banks. Of course, errors in the coding or transmission of the data will result in losses. The subbanding analysis and synthesis processes do not in themselves result in any data loss if properly designed.

There are two places where losses can be intentionally introduced: 1) in the coders, and 2) in the way subband values are treated in terms of maintaining accuracy. Lossy coders deliberately throw out "data" that is irrelevant or is predominantly noise. The signal reconstructed from lossy coders does not (in most cases) exactly match the original signal. This "loss" is by design and in some cases can result in a reconstructed signal that is better than the original. The term "lossy" is unfortunate in that it conjures up thoughts of lost data in scientists' minds when most of what is usually lost is noise.



(a) Uniform bands.



(b) Octave bands.

Figure 3. Subband analysis tree structures.

The second area where losses can be introduced is in the accuracy of subband values. A typical original signal sample size might be 8 bits. When that signal is subbanded, it may take larger sample sizes in the subbands to maintain the accuracy needed for perfect reconstruction. For example, a simple two-channel split will require a minimum of 1/2 bit/sample larger average sample size to maintain accuracy. For a more conventional filter, the sample size requirements are much greater. Although the number of samples doesn't increase over the original signal due to subbanding, the total amount of data increases due to the required accuracy if perfect reconstruction is desired.

An advantage of simple subbanding systems, such as the Walsh-Hadamard transform/subband system (WHT), is that the sample size does not have to increase very much to maintain perfect accuracy. For the two-dimensional, 2x2 WHT, the sample size only increases two bits to maintain perfect accuracy. For more complex subband filters double precision accuracy might be needed to avoid quantization errors (due to truncation or rounding). Section 5 presents a subband system which is even more efficient, in terms of sample size for lossless coding, than the WHT.

For lossy coding the increase in subband value accuracy is not needed because these extra bits would just be dropped in the coding stage. A very effective coding scheme for image data uses simple coarse quantization of the high frequency bands for compression [11].

5. Some Subbanding Algorithms

It can be shown that simple, two-dimensional subband filters can be derived from block transforms [12]. The filters are the basis pictures (formed by outer product of the basis vectors) of the corresponding transform matrix. The transform matrix (in the orthogonal case) is made up of vectors formed from the coefficients of the analysis filters. For the Walsh-Hadamard transform (WHT), with the transform matrix

$$H = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

the two-dimensional filters are:

$$B_0 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix};\ B_1 = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix};\ B_2 = \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix};\ B_3 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}; \qquad (1)$$

To subband an image with these filters, the image component data is broken up into 2x2 blocks and each block is multiplied (inner product) by each of the filters. For example, let D_1 be a 2x2 block (matrix) of luminance pixel values. The low band value, S_0, corresponding to that block is:

$$S_0 = B_0 \cdot D_1 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} d_0 & d_1 \\ d_2 & d_3 \end{bmatrix} = d_0 + d_1 + d_2 + d_3 \qquad (2)$$

The other subband values are:

$$S_1 = d_0 + d_1 - d_2 - d_3$$

$$S_2 = d_0 - d_1 + d_2 - d_3$$

$$S_3 = d_0 - d_1 - d_2 + d_3$$

After these subband values have been calculated, a new 2x2 block of data is read in. If the subband values are kept in a block, then a traditional block transform is obtained. If the subband values are grouped separately (e.g., all the S_0 values in one group), then a subband representation is obtained. The subband values should be scaled (by dividing by 4) and shifted to a positive range if it is desired to view them. If all of the low band (S_0) values are viewed as an image that is 1/2 by 1/2 the original size, then a low resolution, low pass version of the original image is obtained. The higher bands contain edge information.

Since the inverse transform matrix is the same as the forward WHT matrix, the filters for reconstruction are also the same. To get the reconstructed data block, Δ_1, the following equations are used (including the overall transform scaling factor of 1/4 here):

$$\delta_0 = 1/4(S_0 + S_1 + S_2 + S_3)$$

$$\delta_1 = 1/4(S_0 + S_1 - S_2 - S_3)$$

$$\delta_2 = 1/4(S_0 - S_1 + S_2 - S_3)$$

$$\delta_3 = 1/4(S_0 - S_1 - S_2 + S_3)$$

Cascading the algorithm is equivalent to performing a high order subband/transform. For example, performing a four channel subbanding on data that has already been split into four channels results in a sixteen channel split that is equivalent to a 4x4 WHT.

Another subband system from [12] uses the following non-symmetric transform:

$$\boldsymbol{G} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}; \quad \boldsymbol{g} = \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix}. \tag{3}$$

The analysis filters are given by the basis pictures of $\boldsymbol{G}$:

$$\boldsymbol{B}_0 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}; \ \boldsymbol{B}_1 = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix}; \ \boldsymbol{B}_2 = \begin{bmatrix} 0 & 0 \\ 1 & -1 \end{bmatrix}; \ \boldsymbol{B}_3 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}; \tag{4}$$

This system has been characteristic that the low band is a subsampled version of the original. This is useful for chrominance data where an averaging function could create false colors in the low band. This algorithm is also simpler, requiring fewer additions. It is not a symmetric transform, however, and the following filters are used for reconstruction:

$$\boldsymbol{b}_0 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}; \ \boldsymbol{b}_1 = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}; \ \boldsymbol{b}_2 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}; \ \boldsymbol{b}_3 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}; \tag{5}$$

The equations for analysis and synthesis corresponding to (4) and (5) are:

Analysis	Synthesis
$S_0 = d_3$	$\delta_0 = S_0 + S_1 + S_2 + S_3$
$S_1 = d_1 - d_3$	$\delta_1 = S_0 + S_1$
$S_2 = d_2 - d_3$	$\delta_2 = S_0 + S_2$
$S_3 = d_0 - d_1 - d_2 + d_3$	$\delta_3 = S_0$

6. Simulation Results

Some examples of subbanding an image are shown in Figures 4 through 10. Figure 4 shows the original image "Lenna" which was taken from the USC unofficial standard image set. The

image is 512 x 512 pixels (but appears stretched horizontally due to the display system used) and was converted to monochrome from a color original.

Figure 5 shows a four band split of the original using G from Equation (3). The low band in the upper left corner is a subsampled version of the original while the high band in the lower right corner has very little energy. Figure 6 is a close-up of the low band which makes the low resolution artifacts, such as staircase edges, more visible.

The sixteen band split shown in Figure 7 was obtained by cascading two four band splits. The data shown in Figure 5 was subbanded again using the same algorithm. A third cascade of the four band algorithm produced the sixty-four band split shown in Figure 8. A close-up of the low band is shown in Figure 9. The low band is a very low resolution version of the original, but it is usable as a browse picture and is only 64 x 64 pixels in size.

Figure 10 shows a perfect reconstruction of Lenna from a four band split using the inverse transform g from Equation (3).

To investigate the effect of subbanding on lossless compression of images, some subbanded image files were compressed with a public domain program that is based on the Lempel-Ziv coding method. Files subbanded into four, sixteen, and sixty-four bands using both H and G transform matrices were compressed and compared to the original image file size. The original images did not compress well with the LZ coder.

At first, the subbanded values were truncated to 8 bits before compression, resulting in a lossy compression. Since the LZ coder is designed to work on text characters and since the bits that were dropped are expected to contain mostly noise, this approach seemed a logical first step.

Figure 4. Original image of girl, "Lenna."

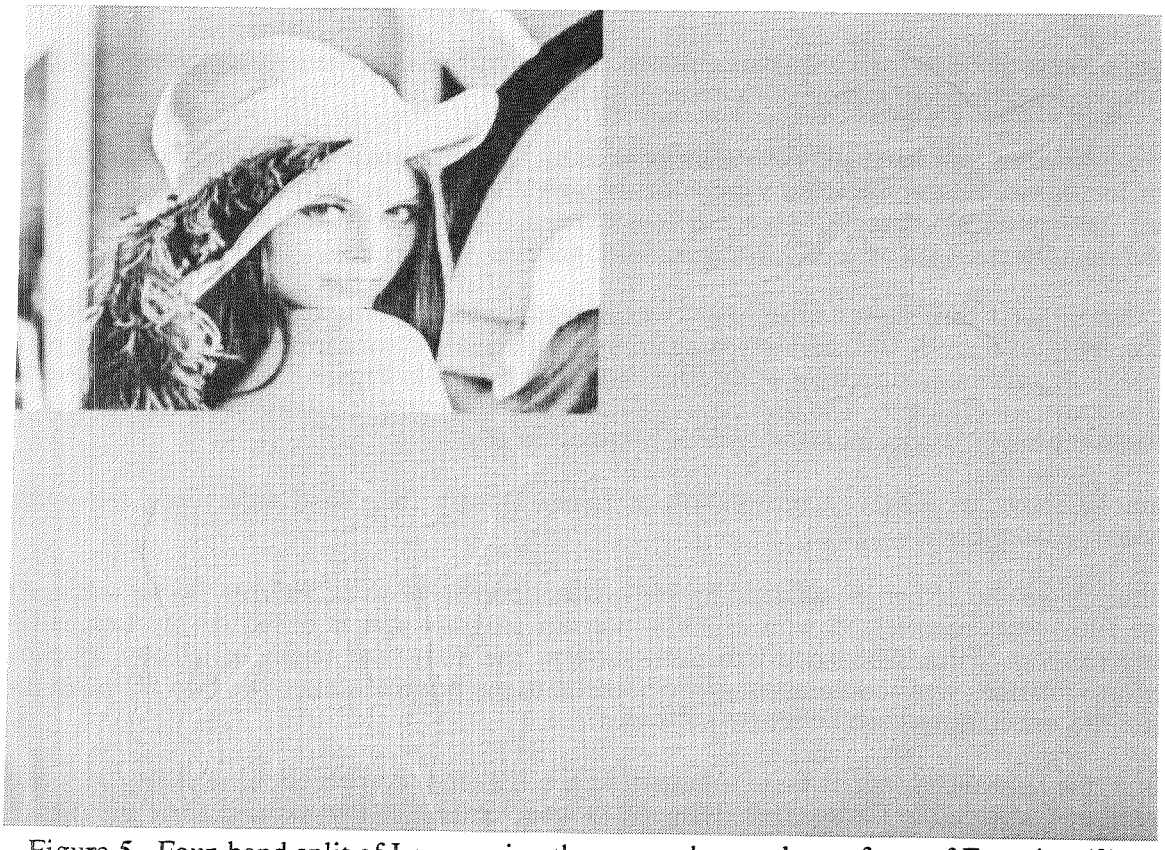


Figure 5. Four-band split of Lenna using the non-orthogonal transform of Equation (3).

Figure 6. Close-up of the low band of Figure 5.

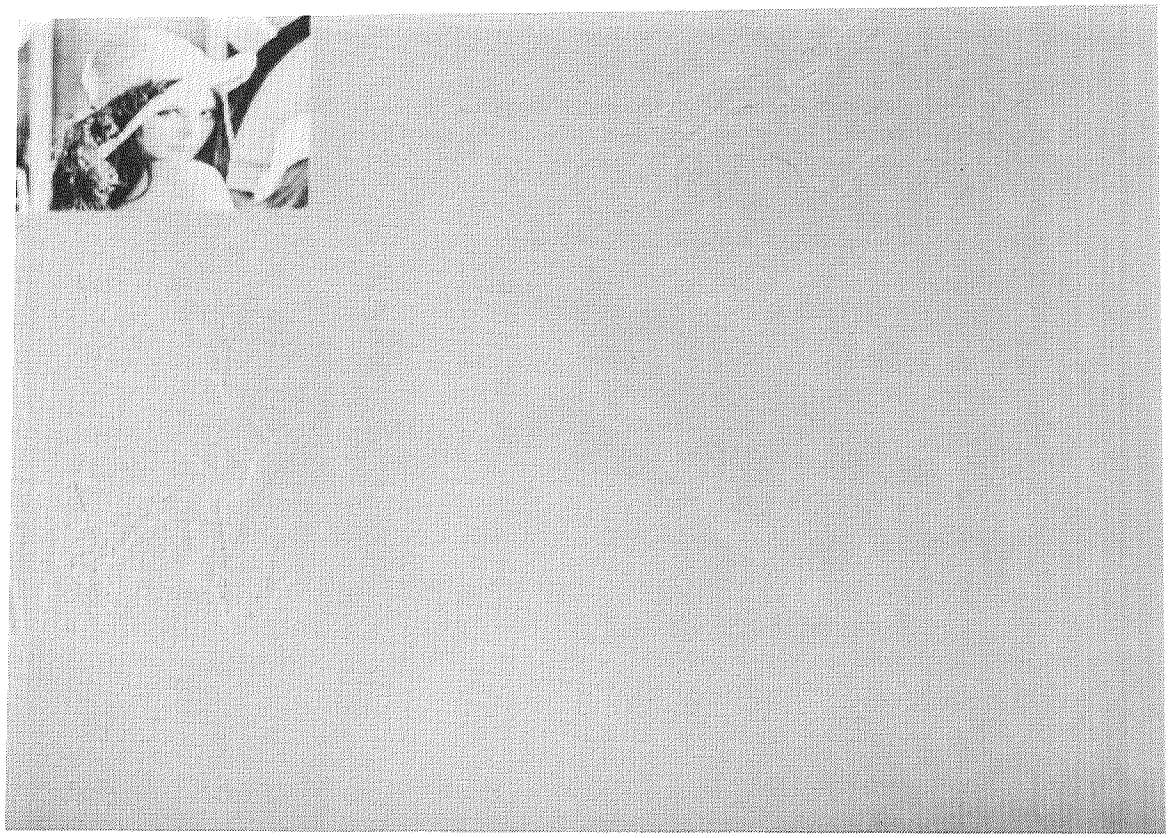


Figure 7. Sixteen-band split of Lenna obtained by applying the transform of Equation (3) to the data shown in Figure 5.

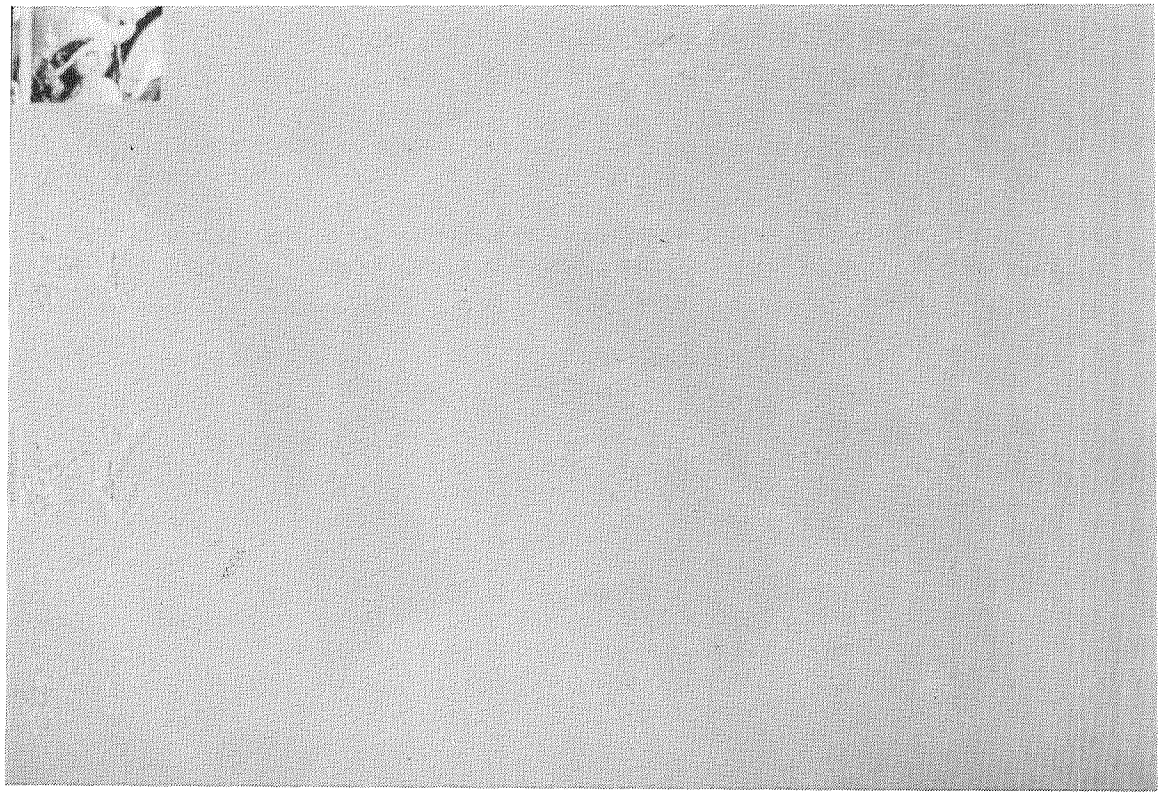


Figure 8. Sixty-four-band split of Lenna obtained by applying the transform of Equation (3) to the data shown in Figure 7.

Figure 9. Close-up of the low band of Figure 8.

Figure 10. Perfect reconstruction of Lenna following a four-band split.

The results are shown in Table 1. The more subbands an image is separated into, the better the compression. The quality of the reconstructed image remains very good since only a small amount of information is lost in truncation. The *H* transform out performed the *G* because when the values were truncated, more signal energy was removed in the *H* transform case (i.e., there is less loss in the *G* case). A comparison of the subband value size between *H* and *G* is shown in Figure 11.

For true lossless compression the full value for each subband must be kept. This is most easily done by storing each value as a 16 bit word. Since the lowest band of a *G* transformed image is still only 8 bits, that band can still be stored in bytes. Using *H*, all bands have the same value size (dynamic range) and so require an extra 2 bits per value for every subband stage.

Since noise is not generally compressible, we would expect the lossless compression ratios to be worse than the truncated ratios of Table 1. Table 2 shows the lossless case for the four band subbanded images. Although the *H* transformed data actually expands instead of contracts, improvements could be easily obtained by packing the data properly before compressing. For example, adding the extra 2 bits per value (for a total of 65 536 bytes) to the four-band truncated compressed file in Table 1 yields a total file size of 263 630 for the Baboon image. The *G* transform does much better, probably due to the 8 bit low band values.

The results of Table 1 show that subbanding image data can improve the effectiveness of LZ compression. The more the data is subbanded, the better the compression ratios. For the truncated values used in these tests, the *H* transform matrix outperformed the *G* because more noise was removed in truncation.

The results in Table 2 show, in a preliminary way, the advantage of the *G* matrix in lossless coding. The *G* transform produces an average of 1 bit/pixel smaller value size than *H* which gives it a head start in compression. The values in Table 2 show a trend, but the bits have not been packed in an efficient fashion. However, computer based methods tend to use byte and 16 bit word size values. Dense packing of bits requires more manipulation.

The interesting case of the Baboon image subbanded with *G* compressing better when the higher bands are stored as 16 bit values than truncated bytes is probably due to random chance. *G* out performs *H* for lossless compression in this simple case. Future research will be into more efficient packing methods.

7. Concluding Remarks

Subbanding image data can be advantageous for browsing and progressive resolution in addition to data compression. Subband systems can be designed to provide perfect reconstruction. The use of short kernel filters allows fast, simple implementations for real-time use or fast retrieval.

Two methods of subbanding image data with simple filters were presented. These methods can be expanded or cascaded to work on larger block sizes easily [12]. Subbanding image data is a promising method of obtaining better performance from lossless coders.

Acknowledgments

This work is based on research that was done in partial fulfillment of the requirements for the Ph.D. degree at the University of Toledo. This work was funded by NASA Lewis Research Center. Thanks to Mr. Paul G. Mallasch of Analex Corp. for coding the algorithms and to Mrs. Linda Gaydosh for preparing the manuscript.

Table 1. Lossy Compression Results using an LZ Coder on subbanded images with all values truncated to 8 bits.

			File Size		
Image	Number of bands	Transform matrix	Original	Compressed	Ratio
Lenna	1	---	262 148	228 196	1.15:1
	4	*H*		166 343	1.58:1
	4	*G*		173 011	1.52:1
	16	*H*		121 418	2.16:1
	16	*G*		142 832	1.84:1
	64	*H*		88 383	2.97:1
	64	*G*		125 100	2.10:1
Baboon	1	———		239 005	1.10:1
	4	*H*		198 094	1.32:1
	4	*G*		202 833	1.29:1
	16	*H*		166 275	1.58:1
	16	*G*		184 933	1.42:1
	64	*H*		136 257	1.92:1
	64	*G*	↓	171 851	1.53:1

Table 2. Lossles Compression Results using an LZ Coder on Subbanded Images Saved as 16 Bit Values Except Low Band of *G* Transform Which Are Saved in Bytes.

				File Size		
Image	Number of bands	Transform matrix	Average pixel size, bits	Original	Compressed	Ratio[a]
Lenna	4	*H*	10	524 292	303 509	0.86:1
		G	9	458 756	254 594	1.03:1
Baboon	↓	*H*	10	524 292	371 813	0.71:1
		G	9	458 756	196 535	1.33:1

[a] Ratio of original 262 148 byte file to compressed.

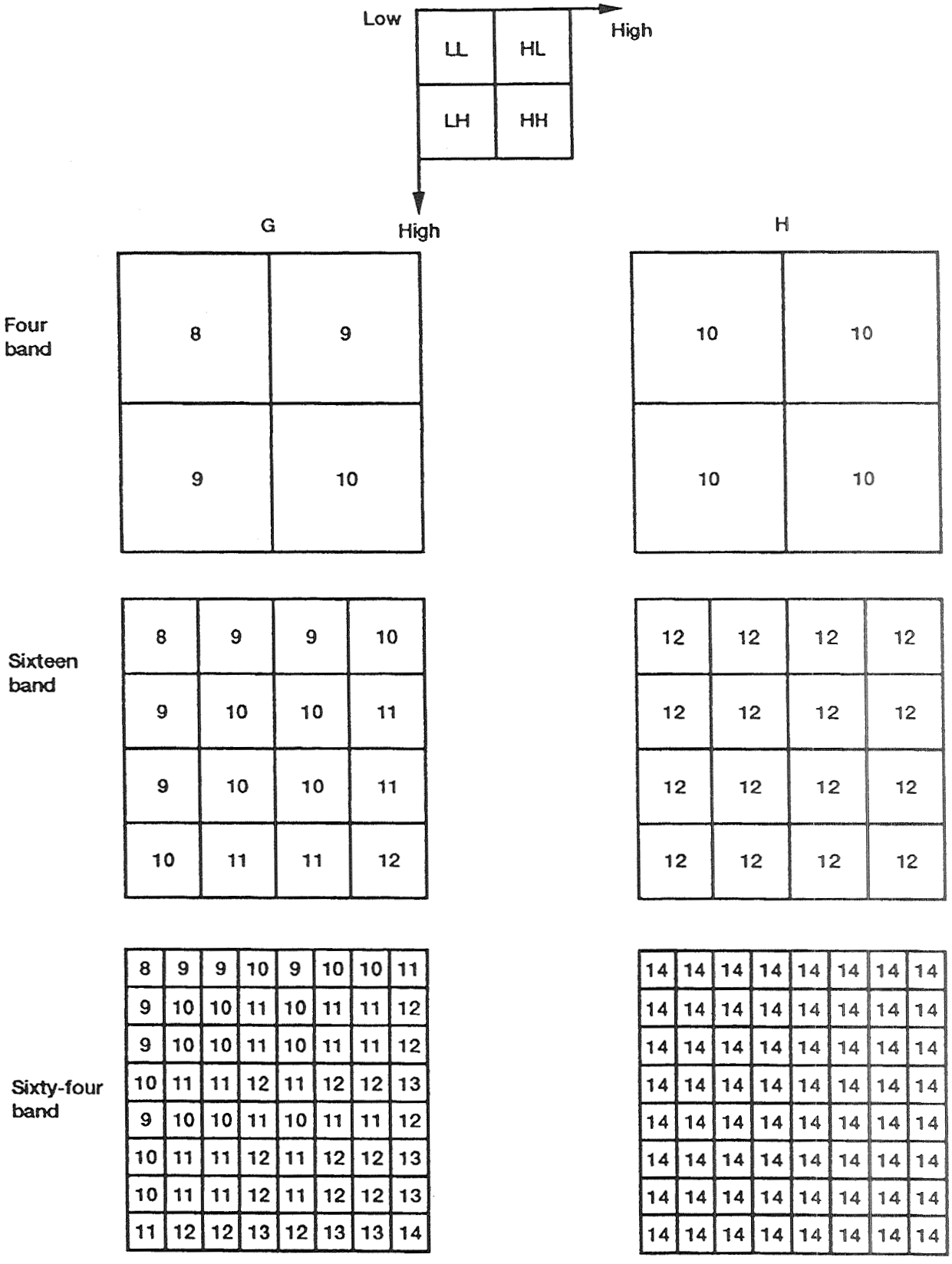


Figure 11. Pixel value size by subband for G and H transforms, cascading the four-band analysis.

References

[1] R. E. Crochiere, S. A. Webber, J. L. Flanagan: "Digital coding of speech in sub-bands," Bell Sys. Tech. J., vol. 55, pp. 1069-1085, October 1976.

[2] A. Crosier, D. Esteban, C. Galand, "Perfect channel splitting by use of interpolation, decimation and tree decomposition techniques," Proc. of Int. Conf. on Information Sciences/Systems, Patras, Greece, pp. 443-446, August 1976.

[3] M. Vetterli, "Multi-dimensional subband coding: some theory and algorithms," Signal Processing, vol. 6, pp. 97-112, February 1984.

[4] J. W. Woods and S. D. O'Neil, "Sub-band coding of images," IEEE Trans. ASSP, vol. ASSP-34, pp. 1278-1288, May 1986.

[5] A. v. Brandt, "Sub-band coding of videoconference signals using quadrature mirror filters," Applied Sig. Processing and Dig. Filtering, Proc. IASTED Int. Symp, Paris, pp. 212-215, June 17-21, 1985.

[6] M. J. T. Smith, T. P. Barnwell, "A procedure for designing exact reconstruction filter banks for tree structured sub-band coders," Proc. IEEE ICASSP-84, San Diego, pp. 27.1.1-27.1.44, March 1984.

[7] R. E. Crochiere, L. R. Rabiner, Multirate Digital Signal Processing, Englewood Cliffs, NJ: Prentice Hall, 1983.

[8] P. P. Vaidyanathan, "Multirate digital filters, filter banks, polyphase networks, and applications: a tutorial," Proc. IEEE, vol. 78, no. 1, pp. 56-93, January 1990.

[9] J. W. Woods (ed), Subband Image Coding, Boston: Kluwer, 1991.

[10] J. S. Lim, Two-Dimensional Signal and Image Processing, Englewood Cliffs, NJ: Prentice Hall, 1990.

[11] A. Al-Asmari, S. C. Kwatra, "Bandwidth compression of HDTV images for communication via satellite," to be presented at the 14th AIAA Conf. on Comm. Sat.

[12] D. Glover, "Subband/transform image coding for lossy and lossless compression," Ph.D. dissertation, The University of Toledo, in preparation.

ADJUSTABLE LOSSLESS IMAGE COMPRESSION BASED ON A NATURAL SPLITTING OF AN IMAGE INTO DRAWING, SHADING, AND FINE-GRAINED COMPONENTS

Dmitry A. Novik
Mail Code 610.3
Universities Space Research Association
Greenbelt, Maryland 20771

James C.Tilton
Mail Code 930.4
NASA Goddard Space Flight Center
Greenbelt, Maryland 20771

Abstract. The compression, or efficient coding, of single band or multispectral still images is becoming an increasingly important topic. While lossy compression approaches can produce reconstructions that are visually close to the original, many scientific and engineering applications require exact (lossless) reconstructions. However, the most popular and efficient lossless compression techniques do not fully exploit the two-dimensional structural links existing in the image data. We describe here a general approach to lossless data compression that effectively exploits two-dimensional structural links of any length. After describing in detail two main variants on this scheme, we discuss experimental results.

1. Introduction

The problem of compressing or efficiently coding various types of data for transmission or archival has attracted much attention recently. Since the trend is continuing in the direction of collecting increasing amounts of space and Earth science data, this should be an important problem for some time to come.

Much of space and Earth science data is represented in image format. This is especially true for data that will be analyzed, recognized, and identified by human beings, with or without the assistance of a computer. The two-dimensional nature of single band images, and the three-dimensional nature of multispectral images, gives this type of data a different nature than one-dimensional data such as alphanumeric text.

Many different scientific and engineering applications utilize image archiving and/or transmission systems. Achieving the highest possible compression factors for the images will help reduce the expense of building and maintaining these systems. Besides space and Earth science applications, many other applications would benefit from compression, including medical and other scientific, industrial, defense, and intelligence image analysis applications; computer assisted image synthesis (e.g. animation); fingerprint and other criminology imaging application; and the publishing industry.

All of these applications require lossless compression of image data, especially in the associated archival systems. It should be emphasized that in many cases the absolute pixel values carry important information in addition to that carried by the general image morphology (as determined by pixel contrast). For example, in CAT or MRI medical images, the morphology represents the anatomy of the body, while the absolute pixel values carry additional information about pathology and disease.

The potential for compression in any meaningful two-dimensional still image data comes mainly from the statistical, spatial redundancy reflecting the two-dimensional structural links between image pixels that form two-dimensional structures. The more two-dimensional structure

contained in an image, the more compressible it will be, since this structure makes the image more dissimilar from a two-dimensional random field.

The amount of compression achieved for an image will depend on how deep and long are the structural links between image pixels, and how well this structure is extracted by the two-dimensional preprocessing of the image compression scheme. The most popular and efficient lossless compression techniques [1], [2], [3], do not effectively exploit these two-dimensional structural links, since they were designed for the compression of one-dimensional messages such as texts.

There are two well known approaches for exploiting the two-dimensional structure contained in still images. The first is two-dimensional predictive coding, in which the value of a pixel is predicted based on knowledge of the directly neighboring pixels previously scanned. Such preprocessing is recommended by JPEG Still Picture Compression Standard [4]. However, the restriction of using only directly neighboring pixels for prediction does not allow the use of any longer two-dimensional structural links between pixels.

The second well known approach for exploiting two-dimensional image structure is block encoding in which the image is artificially divided into k-by-k pixel blocks, and spatial features are extracted from the analysis of each of these blocks [4], [5], [6]. In real images, however, two-dimensional spatial structures generally do not correspond to such artificial blocks, but instead have arbitrary shapes and sizes.

2. General Approach

Our approach to two-dimensional image compression employs a natural splitting of an image into its drawing, shading and fine-grained components. This approach extracts and utilizes two-dimensional structural links of any length between image pixels. Because of this, our method has the potential of achieving better higher compression amounts than other methods that do not effectively utilize these two-dimensional links.

The splitting of an image into drawing, shading and fine-grained components arises naturally from the nature of digital images of natural scenes. A digital image is a matrix of $n_1 \times n_2 = N_0$ pixels, which often is the product of analog-to-digital conversion of a natural scene consisting of objects. Since the objects in the natural scene manifest themselves as connected regions in the two-dimensional scene, any so constructed digital image is a map-image of a number of connected regions that fully cover the image. We will call such an image a digital map-image.

A connected region is an ensemble of pixels of the same type or value, in which it is possible to travel from any pixel in such a region to any other pixel contained in it without intersecting the boundary of the region. For every meaningful digital map-image with N_0 pixels, the number of connected regions, N_{cr}, satisfies $N_{cr} << N_0$. This statistical redundancy is exploited by our approach.

In our approach, the drawing component is a contour map of boundaries between all N_{cr} connected regions of the digital map-image. (How these connected regions are defined is discussed in the next sections.) These boundaries may be represented by a "compound directional edgemap," which we will call a CDEdgemap. A CDEdgemap can be constructed by setting bits as follows:

i) set the first bit if (and only if) the pixel is in a different region from the pixel to the south;

ii) set the second bit if (and only if) the pixel is in a different region from the pixel to the east.

Relationships to pixels to the north and west can be found through reflection. Such two bit region boundary representations are sufficient to select and label separately all N_{cr} connected regions, with four nearest neighbor connectivity. This scheme can be extended to eight nearest neighbors by setting two more bits as follows:

iii) set the third bit if (and only if) the pixel is in a different region from the pixel to the southeast;
iv) set the fourth bit if (and only if) the pixel is in a different region from the pixel to the southwest.

Here the relationships to pixels to the northwest and northeast can be found through reflection.

The shading component is then the list of feature values for each region. The feature value is a common pixel value (or common pixel vector for the multispectral case) for each connected region. This common pixel value can be simply the region mean, or a more complicated compound value, such as a set of texture values. It is sufficient to list these feature values in the order in which the regions are first encountered in a fixed scan order through the image. Knowing the fixed scan order, the drawing and shading components can be combined to produce a possibly lossy reconstruction of the initial digital map-image. Depending on the parameter settings, this lossy reconstruction can be either identical or very similar to the original, or a coarse representation which may be appropriate for image browsing.

The fine-grained component is then the difference between original image and the lossy image reconstruction from the drawing and shading components. Since the drawing and shading components contain much of the structural information, the fine-grained component consists primarily of the region textures.

For further compression, the drawing, shading, and fine-grained components are then processed with a universal adaptive compression technique. We have used the LZW algorithm (available as "compress" on UNIX computers), and two variants of Arithmetic coding.

We have implemented two main variants of our general image compression approach. We will first discuss the variant based on dropping a specified number of least significant bits from the original image data. Later we will discuss the variant based on image segmentation by iterative parallel region growing.

3. Method Based on Dropping m_l Least Significant Bits

In the first variant, the drawing and shading components are defined by dropping a certain number of least significant bits from the original image. In this case, each connected region is the ensemble pixels of the same value (after a specified number of least significant bits are dropped), in which it is possible to travel from any pixel in such a region to any other pixel in the region without intersecting the of the boundary of the region.

Any image with m bits per pixel can be arbitrarily divided into two digital images. Each pixel of one image is the m_l least significant bits of the corresponding pixel in the original image. Each pixel of the other image is the m_h most significant bits of the corresponding pixel in the original image. Note that $m=m_h+m_l$. The splitting into drawing and shading components is performed only on the image with pixels corresponding to the m_h most significant bits in the original image.

The image with the m_l least significant bits is designated as the "fine-grained" component of the original digital image.

It is clear that, under our scheme, the size of the drawing and shading components will depend on the number of connected regions, N_{cr}. N_{cr} is determined by the nature of the original image, by the number of bits, m, in the analog-to-digital conversion used in creating the original image, and the number, m_l, of least significant bits dropped. N_{cr} will generally decrease with a decrease in m and increase in m_l.

Depending of the value of m_l, giving $m_h = m\text{-}m_l$, we will obtain different amounts of image compression. There will exist a value $m_{l,opt}$ that provides the maximum possible image data compression for the combination of all three components (drawing, shading and fine-grain) of a digital image represented with m bits per pixel.

The following is an outline of an algorithm that implements this variant of our image data compression scheme for an m-bit per pixel digital image:

i. select the m_l least important bits from original digital image as its fine-grain component;
ii. separate the image formed by the m_h most important bits into its shading and drawing components, producing a directional edge map and a feature list as described in the previous section;
iii. encode (compress) all three extracted image components by an appropriate universal lossless data compression technique.

The original image is reconstructed by following the steps listed below:

i. decode (decompress) the drawing and shading image components;
ii. construct a region label map by the following process: Give each image pixel the region label RL = (row-1)*ncol + col, where row is the row number, col is the column number, and ncol is the number of columns in the image. Then propagate the lowest region label through each region. Note that there is one unique location in each region where the final labeling matches the original labeling. These locations are also the locations of the first pixel encountered in a line-by-line scan of the image for each region. We call these locations the "seed" pixel for each region;
iii. while scanning through the region label map, fill in "seed" pixels in the reconstructed image from the shading component. Each time a new region is encountered, record the label in a table, and take the next value from the shading component file and place it in that location of the reconstructed image;
iv. propagate the seed pixel value to every pixel in each region using a connected component labeling algorithm;
v. decode (decompress) the fine-grained image component and add it to the reconstructed digital map-image formed by consolidation of the drawing and shading components.

4. Method Based on Image Segmentation

In the second variant of our compression method, the drawing and shading components are defined from an image segmentation produced by an iterative parallel region growing algorithm (IPRG), which is discussed in detail in [7] and [8]. Briefly, the IPRG algorithm is as follows:

i. Initialize the segmentation process by labeling each pixel as a separate region;
ii. merge all spatially adjacent pixels that have identical feature values;

iii. calculate a (dis)similarity criterion between each pair of spatially adjacent regions;
iv. merge pairs of regions that meet the merge constraints;
v. check for convergence. If converged, stop. Otherwise return to step iii.

A dissimilarity criterion is required in step iii. Dissimilarity criterion based on mean-square error and image entropy are described in [8]. For an image compression application, the most natural dissimilarity criterion is to minimize the range of pixel values in each region [9].

Let $MIN(D_k(p))$ be the minimum image value in region p for spectral band k, and let $MAX(D_k(p))$ be the maximum image value in region p for spectral band k.

The dissimilarity function for regions p and q in spectral band k is:

$$F_k(p,q) = MAX(MAX(D_k(p)),MAX(D_k(q))) - MIN(MIN(D_k(p)),MIN(D_k(q)))$$

The dissimilarity function for regions p and q over all bands k, k=1...M is:

$$F(p,q) = \underset{k=1...M}{MAX}(F_k(p,q))$$

The merge constraint employed in step iv is that each merge must be the mutually best pairwise merge for merging regions. This is the same as merge constraint level 1.0 in [7].

Convergence in step iv occurs when there are no further merges that can be performed with the range of pixel values in each region remaining no more than $r_l = EXP(m_e*\ln(2)) - 1$, where m_e is the number of bits per pixel selected for the fine-grained component.

At convergence, the IPRG algorithm produces an edge map coded as described in the General Approach section above. This is the drawing component. It also produces the shading component as a list of feature values (or vectors for multispectral images), one per region, in the order each region is first encountered in a line-by-line scan through the image. The feature value is taken to be the "mid-val" of the regions, defined as the rounded average of the minimum and maximum pixel values in each region (the values are rounded *up* to the nearest integer, if necessary). The mid-val is taken as the feature value, rather than the region minimum or maximum, so that the image reconstructed from just the drawing and shading components will have close to the same average brightness as the original image.

The fine-grained component is defined in such a way that it will consist of positive integers in the range 0 to r_l. This is accomplished by taking the fine-grained component to be the difference between the original image and the image reconstructed from drawing and shading components, plus a bias value equal to $(r_l+1)/2$.

As it was in the first variant, in this second variant the size of the drawing and shading components will depend on the number of connected regions, N_{cr}, produced by the IPRG algorithm. As before, N_{cr} is determined by the nature of the original image, by the number of bits, m, in the analog-to-digital conversion used in creating the original image, and the number, m_e, of bits selected for the fine-grained component. N_{cr} will generally decrease with a decrease in m and increase in m_e.

Similar to the situation for the first variant, there will exist a value $m_{e,opt}$ that provides the maximum possible image data compression for the combination of all three components (drawing, shading and fine-grain) of a digital image represented with m bits per pixel.

Comparing the two variants, with $m_e=m_l$, the value of N_{cr} for the second variant will be no more than the value of N_{cr} for the first variant. Thus, the second variant will generally achieve better compression for the drawing and shading components. However, prior to extensive empirical tests, its is unclear as to which variant will produced more compressible fine-grained components. The second variant, however, requires significantly more computation for encoding than the first variant. However, the decoding process for both variants is virtually identical.

The following is an outline of an algorithm that implements this variant of our image data compression scheme for an m-bit per pixel digital image:

i. select value of m_e, or number of bits, for the fine-grain component;
ii. to help pack the feature values in the shading component in the fewest possible bits, find the minimum value of each spectral band and subtract for each pixel in each band;
iii. perform the IPRG algorithm on the adjusted image from step ii with convergence limit $r_l = EXP(m_e*ln(2)) - 1$. The IPRG algorithm produces the drawing and shading components as a directional edge map and feature list (of region mid-val's);
iv. compute the fine-grained component as the difference between the original image and the image constructed from the regions mid-val's, plus the bias value = $(r_l+1)/2$;
v. encode (compress) all three extracted image components by an appropriate universal lossless data compression technique.

The original image is reconstructed by following the same steps listed for image reconstruction for the first variant, with the exception that the bias value = $(r_l+1)/2$ must be subtracted and the band minimum value added back to each pixel in each band as the final step.

5. Experimental Results

Our compression approaches were tested on image data from:

-> the Landsat Thematic Mapper (TM) instrument (512 line by 512 column by 7 band section);
-> NOAA's Advance Very High Resolution Radiometer (AVHRR) instrument (Global Area Coverage) (512 line by 409 column by 5 band section);
-> JPL's Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) (512 line by 614 column by 16 band section).

For both variants we found the optimal m_l or m_e (number of dropped bits or error bits), N_{CR} (number of connected regions), and CR (compression ratio defined as the number of bytes in the original image divided by the number of bytes in the compressed files). In addition, we compared our results to the compression produced by the Lempel-Ziv-Welch algorithm (UNIX "compress"), by two forms of arithmetic coding (Witten-Neal-Cleary and Adaptive), and by straightforward block averaging.

Our algorithm for straightforward block averaging is as follows:

i. select value of n_i, or size of blocks, for the spatial averaging;
ii. subtract the minimum value of each spectral band from each pixel in each band;
iii. compute region mid-value (average of region maximum and minimum) over n_i by n_i sized blocks and store as feature list;

iv. compute the fine-grained component as the difference between the original image and the image constructed from the blocks of mid-values, plus the bias value = $(r_l+1)/2$;*
v. encode (compress) all feature list and fine-grained image components by an appropriate universal lossless data compression technique.

The numerical results are presented in Tables 1, 2 and 3. The drawing, shading and fine-grained components were each losslessly compressed with one of the lossless techniques, depending on which technique performed best.

Table 1 shows that the natural splitting approaches did not perform very well with the TM image. An alternative view is that the adaptive arithmetic coder did so well it made our approaches look poor by comparison. In contrast, tables 2 and 3 show that our natural splitting approaches did very well compared to all other approaches. Our method based on Iterative Parallel Region Growing posted 47% and 60% improvements, respectively, over the Lempel-Ziv-Welch method on the AVHRR and AVIRIS image data. Similarly, the dropping m_l significant bits method showed gains of 41% and 42%, respectively over the Lempel-Ziv-Welch method on the same data sets. Smaller, but still significant, gains were obtained by both methods over the block averaging method.

The results obtained here encourage us to continue investigating the natural splitting approaches described herein. However, the relatively poor results on the TM image indicate that additional work may be required to truly exploit all the two-dimensional structural links that exist in the data.

References

[1] D. A. Huffman, "A Method for the Construction of Minimum Redundancy Codes," *Proceedings IRE*, Vol. 40, 1962, pp. 1098-1101.
[2] L. Ziv and A. Lempel, "A Universal Algorithm for Sequential Data Compression," *IEEE Trans. Information Theory*, Vol. 23, No. 3, 1977, pp. 337-343.
[3] I. H. Witten, *et al*, "Arithmetic Coding for Data Compression," *Communications of the ACM*, Vol. 30, No. 6, 1987, pp. 520-540.
[4] G. K. Wallace, "The JPEG Still Picture Compression Standard," *Communications of the ACM*, Vol. 34, No. 4, 1991, pp. 31-44.
[5] W. Chen and C. H. Smith, "Adaptive Coding of Monochrome and Color Images," *IEEE Trans. on Communications*, Vol. 25, No. 11, 1977, pp. 1285-1292.
[6] R. M. Gray, "Vector Quantization," *IEEE ASSP Magazine*, No. 4, 1984, pp. 4-28.
[7] J. C. Tilton, "Image Segmentation by Iterative Parallel Region Growing and Splitting," *Proceedings of the 1989 International Geoscience and Remote Sensing Symposium*, 1989, pp. 2420-2423.
[8] J. C. Tilton, "Experiences using TAE-Plus Command Language for an Image Segmentation Program Interface," *Proceedings of the Ninth TAE Users' Conference*, 1991.
[9] J. C. Tilton, "Hierarchical Data Compression: Integrated Browse, Moderate Loss, and Lossless Levels of Data Compression," *Proceedings of the 1990 International Geoscience and Remote Sensing Symposium*, 1990, pp. 1655-1658.

* r_l is computed by finding the maximum range of values in the difference between the original and reconstructed image, finding the number of error bits, n_e, required to represent this error and calculating $r_l = \mathrm{EXP}(n_e * \ln(2)) - 1$.

Table 1. Landsat Thematic Mapper (TM) Image

Washington, DC image; 7 spectral bands, 512 lines by 512 columns
1 byte pixels, 1,835,008 bytes in original

Method*	optimal n_j, m_l or m_e	N_{CR}	# bytes compressed	CR	CR ADAP-CR	CR BA-CR
LZW	---	---	1,027,045	1.79	0.83	0.88
WNC	---	---	1,057,307	1.74	0.80	0.86
ADAP	---	---	848,945	2.16	---	1.07
BA	4	16,384	*(WNC: param.)* 905,017 *(ADAP: feat. & res.)*	2.03	0.94	---
DLSB	1	120,431 *(average)*	*(LZW: param.)* 952,580 *(ADAP: feat. & edge)* *(Bit-Pack: residual)*	1.93	0.89	0.95
IPRG	5	4,828	*(WNC: param. & feat.)* 890,743 *(ADAP: edge & res.)*	2.06	0.95	1.02

Table 2. Advanced Very High Resolution Radiometer (AVHRR) Image

5 spectral bands, 512 lines by 409 columns
2 byte pixels, 2,094,080 bytes in original

Method*	optimal n_j, m_l or m_e	N_{CR}	# bytes compressed	CR	CR LZW-CR	CR BA-CR
LZW	---	---	1,121,165	1.87	---	0.92
WNC	---	---	1,229,692	1.70	0.91	0.84
ADAP	---	---	1,292,551	1.62	0.87	0.80
BA	8	3,328	*(WNC: param.)* 1,022,751 *(ADAP: feat.)* *(LZW: residual)*	2.04	1.09	---
DLSB	3	60,177 *(average)*	*(LZW: param.)* 795,258 *(ADAP: all others)*	2.63	1.41	1.29
IPRG	6	7.037	761,864 *(ADAP: all files)*	2.75	1.47	1.35

* LZW => Lempel-Ziv-Welch method, WNC => Witten-Neal-Cleary Arithmetic Coding, ADAP => Adaptive Arithmetic Coding, BP => Bit Packing, BA => Block averaging method, DLSB => Delete Least m_l Significant Bits method, IPRG => method based on Iterative Parallel Region Growing segmentation.

Table 3. Airborne Visible/Infrared Imaging Spectrometer Image

Moffet Field, CA image; 16 spectral bands (out of 210), 512 lines by 614 columns
10,059,776 bytes in original

Method*	optimal n_i, m_l or m_e	N_{CR}	# bytes compressed	CR	CR / LZW-CR	CR / BA-CR
LZW	---	---	6,799,419	1.48	---	0.89
WNC	---	---	6,883,323	1.46	0.99	0.88
ADAP	---	---	6,879,958	1.46	0.99	0.88
BA	4	19,712	*(WNC: param. & feat.)* 6,068,613 *(ADAP: residual)*	1.66	1.12	---
DLSB	5	71,666 *(average)*	*(ADAP or BP: residual)* 4,794,906 *(ADAP: all others)*	2.09	1.42	1.27
IPRG	7	16,267	*(WNC: param. & feat.)* 4,254,146 *(ADAP: residual)*	2.36	1.60	1.43

* LZW => Lempel-Ziv-Welch method, WNC => Witten-Neal-Cleary Arithmetic Coding, ADAP => Adaptive Arithmetic Coding, BP => Bit Packing, BA => Block averaging method, DLSB => Delete Least m_l Significant Bits method, IPRG => method based on Iterative Parallel Region Growing segmentation.

ENTROPY REDUCTION VIA SIMPLIFIED IMAGE CONTOURIZATION

Martin J. Turner
University of Cambridge Computer Laboratory
New Museums Site, Pembroke Street
Cambridge. CB2 3QG. ENGLAND

Abstract. The process of *contourization* is presented which converts a raster image into a set of plateaux or contours. These contours can be grouped into a hierarchical structure, defining total spatial inclusion, called a *contour tree*. A *contour coder* has been developed which fully describes these contours in a compact and efficient manner and is the basis for an image compression method. Simplification of the contour tree has been undertaken by merging contour tree nodes thus lowering the contour tree's entropy. This can be exploited by the contour coder to increase the image compression ratio. By applying general and simple rules derived from physiological experiments on the human vision system, lossy image compression can be achieved which minimises noticeable artifacts in the simplified image.

1. Contourization -- The Basic Principles

Contourization is the creation of an image as a contoured landscape with a one-to-one or one-to-many mapping from pixel values to pseudo height values. An image can be fully described and represented as a set of contours where each contour is a region whose perimeter pixels' values have similar magnitude. Contours can have within their boundaries other contours but no boundary edge pixel is allowed to be part of more than one contour boundary.

A contour description of an image offers two advantageous features over a standard raster image format:

- Image operations can be carried out on contours in one step rather than at the pixel level, giving certain image manipulations potential speedups.
- This representation potentially allows a more compact description format for reduced storage and transmission time.

The rest of this paper is divided into two main sections, firstly a coding method to represent a contour description as compactly as possible and then a new method for lossy compression by exploiting this representation.

2. Contour Description and Coding

A contour description of an image is not a new idea and many related edge description techniques have been studied. These descriptions have all been used, in the past, as aids to image coding. Presented is an up to date implementation which performs as well if not better than most of the current image representations.

2.1 Contour Description

To describe a contour fully three distinct pieces of information are sufficient:[1]

Start Location of the contour in the x-y space of the raster image. This is defined as the top left hand corner of the contour, as in a raster scan of the original image.

Height Value of the contour's perimeter.

Boundary Description of the contour. This is a representation of the shape of its perimeter.

The process of extracting contour information from images can be achieved in many ways which divide into two broad methods, either linear with the scan remembering a current list of half seen contours or processing contours as they are first seen, in a raster scan. The latter method was developed for coding efficiency reasons and a contour following technique commonly called a Backtracking Bug Follower was used as described in [8].

2.2 Contour Coding

Once the contours have been isolated they can be coded producing the three streams of information as stated above.

Preceding these streams of information is a very small header containing three items:

- *Horizontal Size* of the original raster image in pixels.
- *Vertical Size* of the original raster image in pixels.
- *Magic Number* specifying which version coded this image. This is purely for development purposes so versions can be compared.

As an aid towards efficient coding, contours are coded in the order they are first seen in a raster scan sense. This means that when the decoder is dealing with the $(n + 1)$th contour it knows all about the previous n contours and thus all about every pixel's value before the $(n + 1)$th contours' start location. This ordering has the advantage over the obvious alternative of ordering by height as it aids both the coding of start locations and boundary descriptions.

The next three parts deal with coding the three pieces of information for the $(n + 1)$th contour.

2.2.1 Start Locations. The start location is coded as the next possible offset from the last contours' start location, reduced by the number of impossible start location known by the decoder at the time of decoding. Impossible start locations known by the decoder consist of the set of boundary points of the previous n contours. Figure 1 shows a case where contours W and X have been coded and any pixel after contour X's start location not on a known boundary is a potential start location for the next contour.

Also known by the decoder is a maximum offset the start location can be, which consists of the first pixel which cannot belong to any of the previous n contours. This pixel must then belong to an as yet undefined contour as all pixels in the image are in a contour.

---

[1] This specific choice for describing a contour is possibly not the best way and is not justified in this paper.

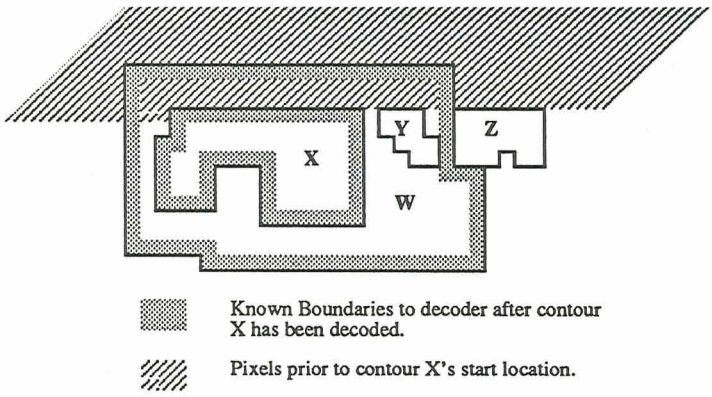


Figure 1. Contour decoding just before contour Y is to be decoded.

The resulting stream has been observed to be highly skewed and to exploit this high probability content it is passed through an Adaptive One Level Conditional Arithmetic Coder.[2]

2.2.2 Height Values. Intensity values in an original raster image often have an inherent skew exploited by many prediction based image compressors, the most common being the Differential Pulse Code Modulation (DPCM) family of coders. This effect is drastically reduced when only one height value per contour needs to be coded but it is still possible to take advantage of it. Similarly encoding the difference from previous contour height usually reduces the information content, so difference values are passed to an Adaptive One Level Conditional Arithmetic Coder.

2.2.3 Boundary Description. The Boundary Description of a contour can be exceedingly complicated but must be represented exactly. A clockwise cyclic route from the start location around the boundary and returning to the original start location gives a simple path. This code is also comma free, by imposing that you have to end up at the start location. Using a simplified 4-way connected version of Freemans line code [3], two bits per movement step describe this route fairly compactly, see Figure 2.

On a boundary description there will be as many "north" moves as "south" moves and as many "east" moves as "west" moves. As a boundary description involves a clockwise walk if the relative moves are described in terms of "forward", "backward", "right" and "left" there should be more "right" moves then "left" moves and more "forward" moves then "backward" moves, which can be exploited, so this later format is adopted.

There are two subtly different ways of going around the boundary, either through the centre of pixels or around the edge of pixels. Both of the methods have their own tradeoffs:

Movements Through Centres (MTC):

- All four directions are possible, so a maximum of two bits per movement step are required.

---

[2] An Arithmetic Coder is a probability coder which works on similar principles to the Huffman Coder except with greater efficiency. When coding a symbol a simple table of probabilities for all symbols has to be present for both coder and decoder. The resulting coded size for any particular symbol is smaller the larger its probability of occurrence. The basic design of the Adaptive Arithmetic Coder implementation by Bell [1] was used, changed slightly by adding one level of conditionality, which allows it to deal with conditional probabilities.

- A single pixel contour has to be treated as a special case and a simple system is to code it as a "north" move, which cannot occur as all pixels prior to the start location are represented in previous contours.

- Certain boundaries cannot be represented and have to be represented as two contours. These occur when the cyclic boundary path has to return through the start location before the end. This is demonstrated in Figure 3.

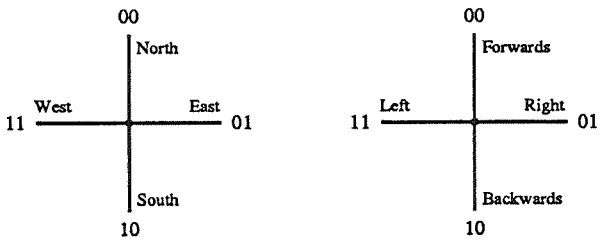


Figure 2. Simplified 4-way Freeman Direction Code, with two alternative labellings.

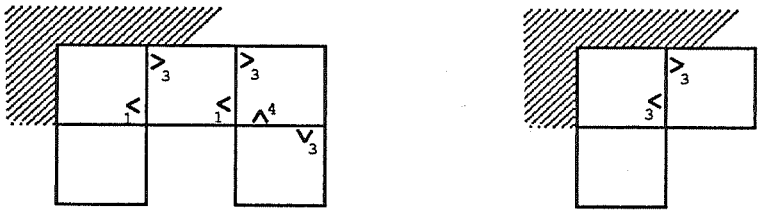


Figure 3. Examples of two simple contours which have to be coded as two contours under MTC. The start locations are in the top left and the numbers indicate the number of choices at each movement stage.

Movements Along Edges (MAE):

- As you never return along the edge you have just come from only three of the possible directions are valid at any point so a maximum of $\log_2 3 \approx 1.585$ bits per movement step are required.

- The difference between the number of movement steps varies greatly with MTC. Boundary descriptions increase in size greatly for small contours; 61% increase for a contour of two pixels, and reduce in size for large boundaries.

- All possible contours can be specified and the single pixel contour does not need to be treated as a special case. Although the single pixel now has to be represented by two[3] movement steps.

---

[3] Note not four steps as all pixels prior to the start location are known meaning that the two movement steps from top left to top right and from bottom left to top left have to be made so are not coded.

Both of these boundaries paths can have their possible movement choices restricted by six external factors:

1. Pixels prior to the start location are all known not to be included in this contour.

2. Horizontal and vertical edges of the full raster image are known.

3. All the contours previously coded will be known by the decoder at this stage and any whose boundary abuts the boundary for this contour will impose movement restrictions.

4. Whilst describing the boundary description this perimeter line can not cross itself so the previous part boundary so far described also acts as a movement restriction.

5. As a clockwise route is taken to define the boundary description pixels on the outside of the part boundary so far described cannot subsequently form part of the remaining section of the boundary.

6. If during the clockwise route a neighbouring pixel is known to be on the contour any pixel which requires a further rotation to the right cannot thus be on the next step.

The following graph in Figure 4 compares the number of bits required to represent the 36 rectangular contours, whose sides are less than seven, with restrictions imposed by factors 1, 4, 5 and 6. The identity line $y = x$ graphically displays when one method gives a higher compression ratio then the other, and the line $y = x(\log_2 3)/2$ is the asymptotic line approached as contour sizes increase to infinity. Figure 5 shows an example of a contour being coded by the two methods.

The cases where MAE gives a coding advantage over MTC, are those where a high level of compression already exists. This means that when an image which gives a high compression ratio with MTC will in general give a higher compression ratio with MAE, but when there is a poor compression ratio with MTC there is a resulting poorer compression ratio with MAE. MTC gives an exceedingly low expansion of random data. To create a universal coder which works well on a wide range of images MTC was used.

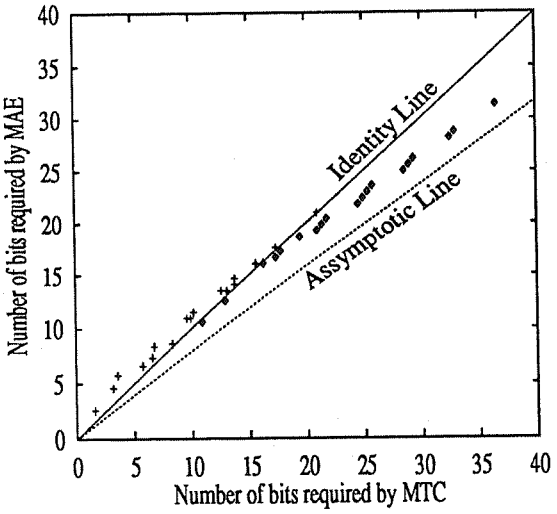
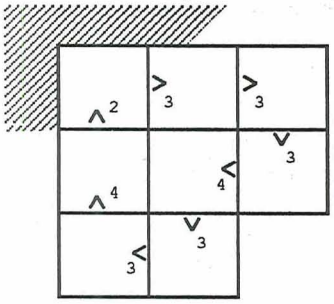


Figure 4. Plot comparing compactness of MTC and MAE for rectangular contours.

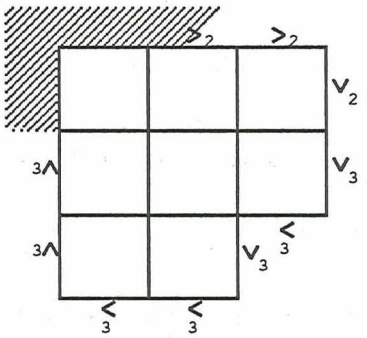
Movement Through Centres

Start at top left pixel heading North.



Movement Along Edges

Start at top right of top left pixel heading East.



```
Path : R, F, R, R, L, R, R, F.

Choices: 3, 3, 3, 4, 3, 3, 4, 2.

# bits: log(3)+log(3)+log(3)+log(4)+
        log(3)+log(3)+log(4)+log(2) = 12.925.
```

```
Path : F, F, R, F, R, L, R, F, R, F.

Choices: 2, 2, 2, 3, 3, 3, 3, 3, 3, 3.

# bits: log(2)+log(2)+log(2)+log(3)+
        log(3)+log(3)+log(3)+log(3)+
        log(3)+log(3) = 14.095.
```

Path gives a list of relative movement steps from "forwards", "backwards", "right" and "left".
Choices gives the total number of different movement directions known to the decoder.
bits calculates the total number of bits required to code this path.

Figure 5. Example of the coding of a simple eight pixel contour using the two alternative methods.

An ideal solution would be to merge the two coding schemes but any simple bookkeeping system either is inefficient or causes too much overhead. Also any merging system would reduce the efficiency of any subsequent probabilistic coding of the stream.

The stream of movement steps is still a very inefficient method of representing a boundary as many movement patterns constitute invalid boundary descriptions.

To reduce this inefficiency a two stage process was designed:

The first stage uses as much of the information known by the decoder to restrict the allowable movement directions as listed above. A *Direction Mask* is calculated which flags allowable movement directions.

The second stage passes the movement steps through an Arithmetic Coder with a Dynamic Markov Model whose initial model has been set up to detect all impossible movement paths up to a length of eight. This modelling system for the Arithmetic Coder is described in Horspool [6] and allows a fast adaptation of movement paths. Common paths are given a higher probability over further impossible or uncommon paths. The mask from stage one allows current impossible movement directions to be discounted.

2.3 Storage

The final image is represented as three separate files or streams and a small header. All of these files can be concatenated together to minimise storage overheads or kept as separate files which incurs only a slight overhead.

3. Contour Trees -- A Structured Approach

The list of contours can be very large and at present no ordering system has been imposed. For coding, to aid compaction, a simple first seen order in a raster scan is very useful but for image manipulation it is limiting. A random pixel access operation is as costly as an ordered list search.

To combat this a hierarchical structure called a contour tree was designed which stores those contours totally enclosed by another contour as its children. The structure commences with an imaginary contour defining the image boundary and having all image contours as its children. Siblings can be arranged in some order, first seen in a raster scan is a simple choice. This makes random pixel access times equivalent to a hierarchical order list search. The structure is similar in some senses to a quadtree but its structure is dictated by the original image rather than by arbitrary division.

For image manipulation operations each contour tree leaf can be allocated certain properties, for example; size, bounding boxes and previous historical operations carried out, have proved useful.

4. Contour Merging to Aid Lossy Compression

Given a contour tree, for each contour we can assign a rating of noticeability according to its size, actual intensity and neighbourhood. It can then be decided if two contours when merged together, creating a joint contour of equal mean intensity, will be noticed by a human observer up to some fidelity criteria. This gives us a way of calculating a merging threshold value.

By repeated merging until all contours cannot merge any further without going over this fidelity criteria, the resulting contour tree is pruned. Each node is usually more complicated but contour coding should result in a higher compression ratio.

This is akin to work carried out by Samet [9] on quadtrees where under certain conditions the quadtree is repeatedly pruned giving a simplified tree. The concept is very similar except that the choice of pruning is gained from the actual structure of the image rather than the local pixel neighbourhood.

The non-deterministic physiological problem is deciding what criteria constitutes the level of noticeability for the merging of two contours.

The current criteria follows these rules:[4]

1. The size of the larger contour is irrelevant as the eye concerns itself with local changes. It has been observed that details attract attention.

---

[4] General rules regarding the physiological and psychological aspects of the eye were extracted mainly from Boff [2], with countless exceptions from other sources mainly the Physiology Department, Cambridge.

2. The size of the smaller contour is critical and its area is linear with the possibility of being recognised as a separate area. Experiments have shown that noticeability of an object in the human eye is proportional to its area.

3. The intensity difference over background intensity ratio is treated linearly with the possibility of merger. It was discovered that average light intensity seems to follow Weber's Rule.

$$\frac{\delta I}{I} = K \text{ a constant}$$

This means that there exists an n under certain conditions such that a change in intensity is only noticed when the change in intensity is greater than $n\%$. Experimentally this has been shown to be accurate only for average light intensities and short exposure times.

4. The maximum "historic" difference of contours intensity values is considered, not just the current value for each contour. This means as a contour expands its constituent contours' intensity are still considered.

5. The number of times either of the contours have already merged before being considered for this new merge. This affects the threshold value by a linear step. This rule was introduced to compensate for effects caused by the iterative way the merging algorithm is implemented.

A single threshold value is specified and two contours are merged together when the deciding algorithm returns a smaller number.

The resulting image can then be passed to the previously described contour coder.

5. Operational Costs

The two operations of contour merging and contour coding are quite separate. Contour merging operations purely pre-filter the image before being transmitted or stored, as shown in Figure 6. The critical time operations then are the contour encoding and decoding.

The costs for contour coding are linear with the size of the image with the decoder requiring less than four random data access per image pixel. The three streams can be coded and transmitted separately splitting the work required to three linear probability coders. The coding process as it presently works requires frame store for the entire picture to be held both at the encoder and decoder which with present systems is not an unacceptable amount of storage.

The costs for the contour merging are more severe being $O(n\langle l \rangle)$, where n is the number of contours and $\langle l \rangle$ is the average length of the resulting contour boundary descriptions.

6. Results

For many computer generated images, even rendered and ray-traced images, the resulting compression ratio of the original is reasonably high. The main thorn in the side is with real images, often captured with cameras or scanners, which introduce their own biases and noise. It is these images, whose contour tree description are very flat with an enormous number of small contours, that will be studied in this section.

6.1 Test Images

Four non-synthetic images are presented, all are stored at 8bpp gray scale:

balloons a famous test image picture, showing a girl within a cascade of smoothly shaded balloons.
mandrill the famous test image picture of a monkey's face.
escher a 300dpi scanned on an Apple Scanner, sketch drawing by M.C. Escher.
garden an image captured by a video camera.

These pictures are shown in Figure 7. Height values in the contour description are mapped directly from the actual gray scale intensity values.

6.2 Analysis

The size of the contour trees for each image with the number of contours at each level of the contour tree are shown in the second table. These images have very little structure, the "mandrill" contour tree consists of over 94% single pixel contours. The first table and graph compares compression ratios of the contour coder (CC) with an Adaptive One Level Conditional Arithmetic Coder (AC) as described in Witten [13], the LZT variant in the Lempel-Zif family as first presented in Tischer [10] and a Quadtree Coder (QTC) described in Turner [11]. Also shown is the theoretical compression ratio from entropy calculations using Shannon's formula:

$$E = -\sum_{0}^{N-1} p_n \log p_n$$

where p_n is the probability of pixel value n, in the range 0...N-1, of occurring.

Second order and third order entropy range over pairs and triples of pixel values, and Diffs is first order entropy over pixel value differences.

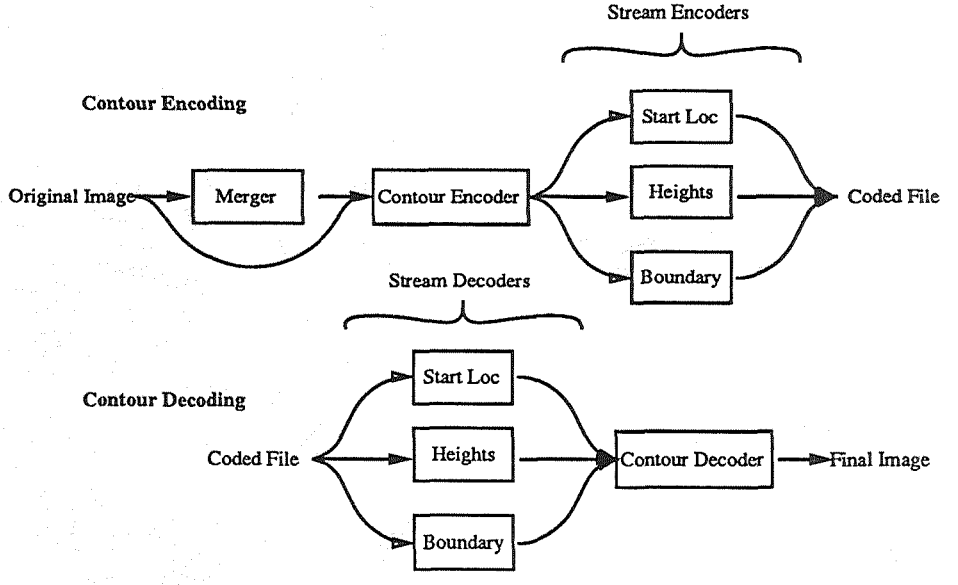


Figure 6. Schematic diagram of the contour coding with and without application of the premanipulating merging algorithm.

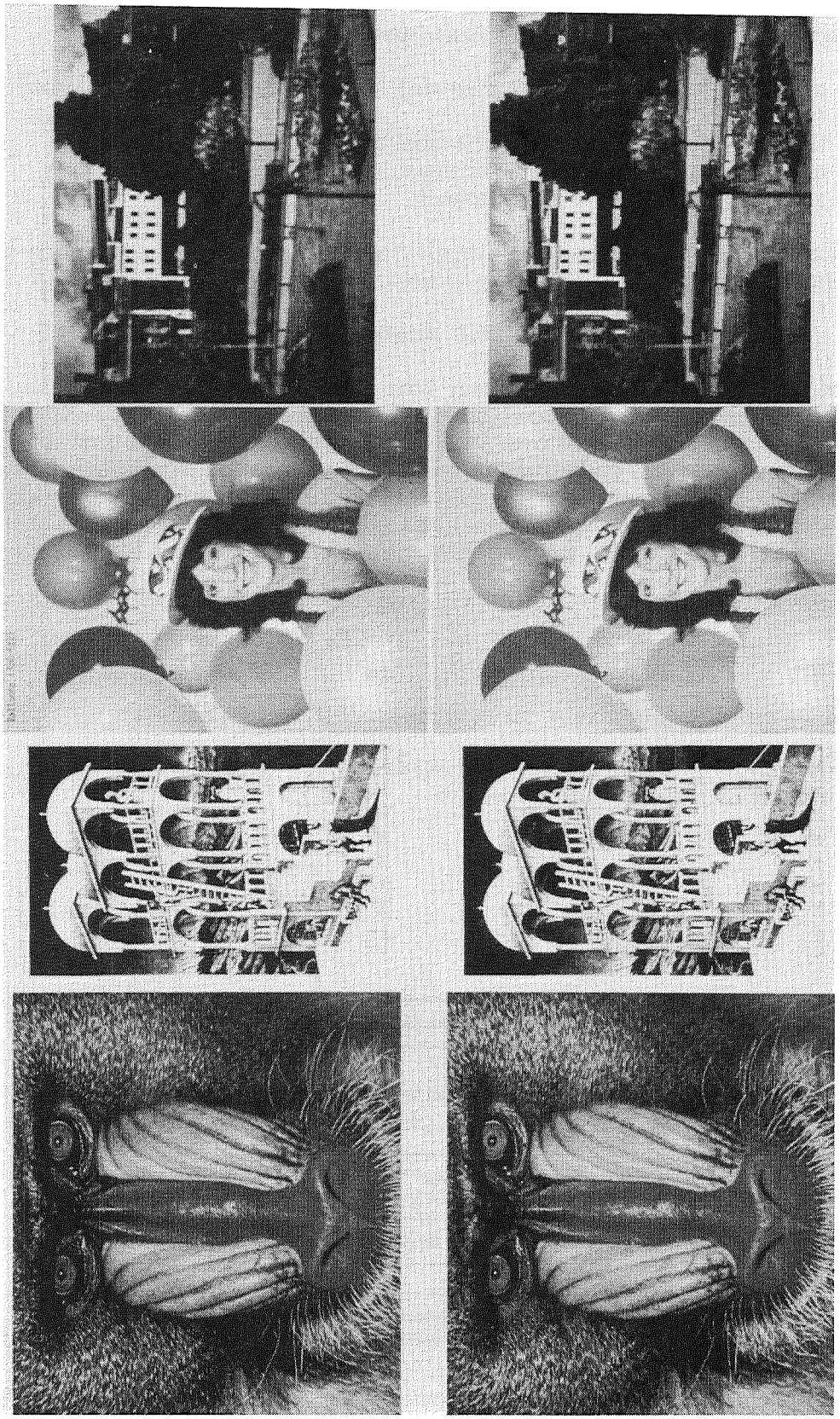


Figure 7. Comparison of originals with *excellent* version: normal colour map.

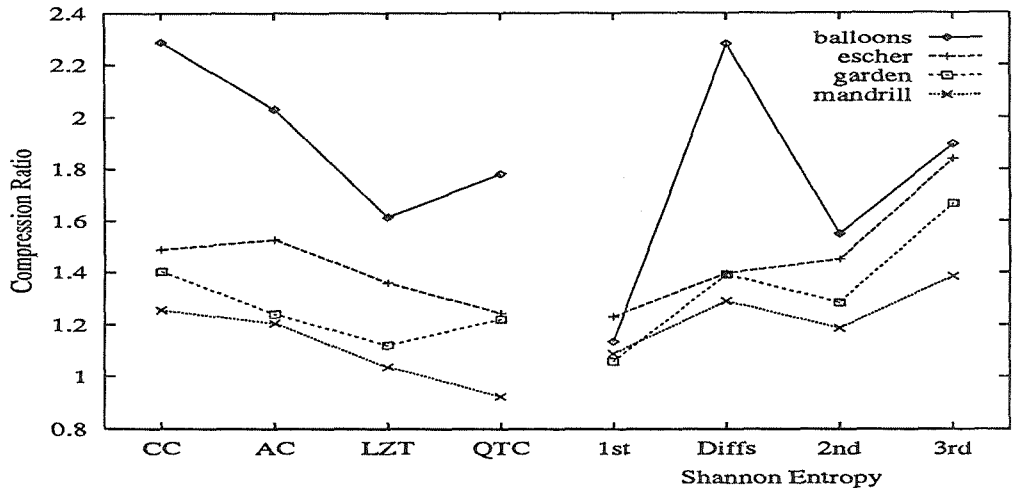


Image	*CC*	*AC*	*LZT*	*QTC*	*1st Order Entropy*	*1st Order Diffs*	*2nd Order Entropy*	*3rd Order Entropy*
balloons	2.286	2.030	1.614	1.781	1.136	2.280	1.550	1.896
escher	1.488	1.527	1.358	1.244	1.231	1.396	1.450	1.839
garden	1.402	1.240	1.121	1.221	1.060	1.391	1.284	1.667
mandrill	1.254	1.205	1.037	0.924	1.090	1.290	1.187	1.384

To define quality of an image simplified by the contour merging algorithm the term *excellent* was devised which has the simple criteria that a subjective observer cannot tell the original from the merged version when shown separately. These images are shown in Figures 7 & 8. Compression results achieved with this "excellent" criteria. The next table quantifies the number of contours and compression ratios achieved, in the original (first line) and "excellent" images (second line).

Image	*Size:pxls*	*Comp. Ratio*	*No. Contours*	*Hierarchical Depth*		
balloons	198664	2.286	85930	84652	1278	
		5.771	5892	4925	966	1
escher	138632	1.488	91546	1	91199	346
		4.715	8443	1	7299	1143
garden	49160	1.402	40613	40498	116	
		4.356	3107	2887	219	1
mandrill	245768	1.254	231086	231086		
		3.930	19744	17823	1921	

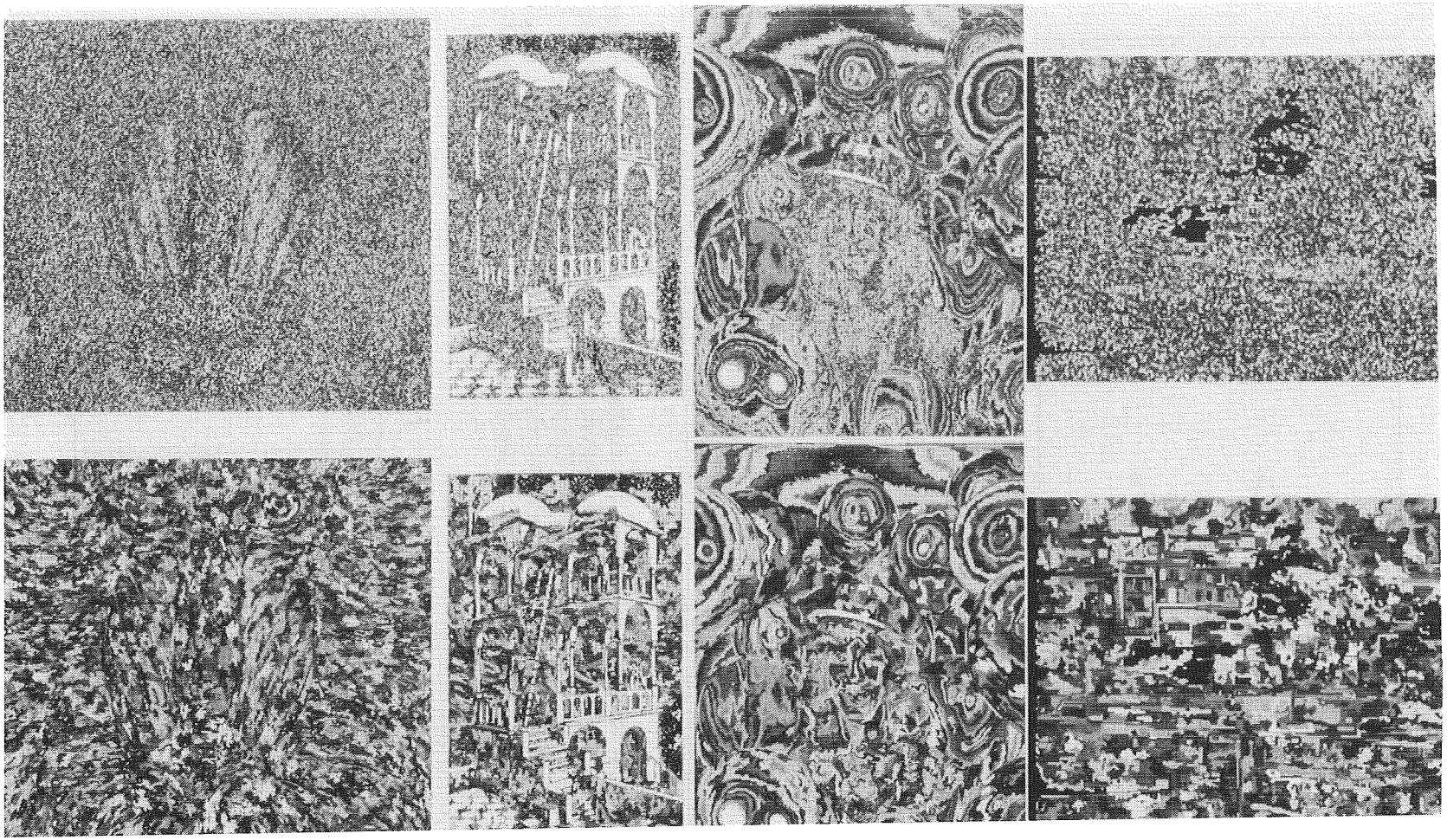


Figure 8. Comparison of originals with *excellent* versions: random colour map.

6.3 Visual Artifacts

Those images which are "excellent" canstill have up to one fifteenth the number of contours which with the contourcoder gives compression ratios between 4:1 and 6:1. When larger threshold values are given to the contour merging algorithm artifacts start to be seen. The main artifacts are:

Loss of Highlights as in the flattening of the pupil of the mandrill's eye, shown in Figure 9.

Loss of Connectivity between regions which the mind psychologically fills in. This can be seen in the top railing on the image "escher" in Figure 10. Merging operations on the left and right of the already partially disconnected railing separates it even further.

Blotching occurs due to a high level of merging. Noticeable in areas of randomness an example is in the flower bed of the image "garden", as shown in Figure 11.

False Contouring occurs when smoothly shaded areas group into regions. Noticeable on low quality printers which add to the effect, due to intensity quantization.

Two features which the merging algorithm exploits with its simple set of rules:

Edges are Sharpened up which according to Marr [7] are particularly recognisable in the human brain.

Texture Shape via the contour merging algorithm is detected at the low level and the merging process flows with it. This allows a way of simplifying and describing areas such asgrass and hair.

7. Conclusions

Demonstrated was a new technique in lossy image compression which offers an alternative to other currently used methods. When combined with an up to date contour coder the initial results are very promising and demonstrate that its use is viable.

References

[1] T. C. Bell, J. G. Cleary, I. H. Witten, *Text Compression*, Prentice Hall Advanced Reference Series.

[2] K. R. Boff and J. E. Lincoln (ed), *Engineering Data Compendium, Human Perception and Performance*, Harry G. Armstrong Aerospace Medical Research Lab.

[3] H. Freeman, "On the Encoding of Arbitrary Geometric Configurations," *IRE Transactions on Electronic Computers*, June 1961.

[4] R. C. Gonzalez and P. Wintz, *Digital Image Processing*, Second Edition, Addison-Wesley Pub. Advanced Book Program.

[5] D. N. Graham, "Image Transmission by Two--Dimensional Contour Coding," *Proceedings of the IEEE*, Vol. 55, No. 3, March 1967.

[6] R. N. Horspool and G. V. Cormack, "Dynamic Markov Modelling -- A Prediction Technique," *Proc. International Conference on the System Sciences*, Honolulu, HA, January 1986.

[7] D. Marr, *Vision*, Freeman.

[8] W. Pratt, *Digital Image Processing*. Second Edition. Wiley-Interscience, ISBN 0-471-85766-1.

[9] H. Samet, "Data Structures for Quadtree Approximation and Compression," *Comms. of the ACM.*, Vol. 28, No. 9, Sept. 1985.
[10] P. Tischer, "A Modified Lempel--Ziv-Welch Data Compression Scheme," *Australian Computer Science Communications*, Vol. 9, No. 1, pp. 262--272.
[11] M. J. Turner, "ICEBERG - Gray Scale Image Compression Scheme Part II," *Computer Science Tripos 1990*, Dissertation, Cambridge University, UK.
[12] T. A. Welch, "A Technique for High Performance Data Compression," *IEEE Computing Society, Computing*, Vol. 17, No. 6, June 1984.
[13] I. H. Witten, R. Neil, J. G. Cleary, "Arithmetic Coding for Data Compression," *Comms. of the Assoc. for Comp. Mac.*, Vol. 30, No. 6, June, pp. 520--540.

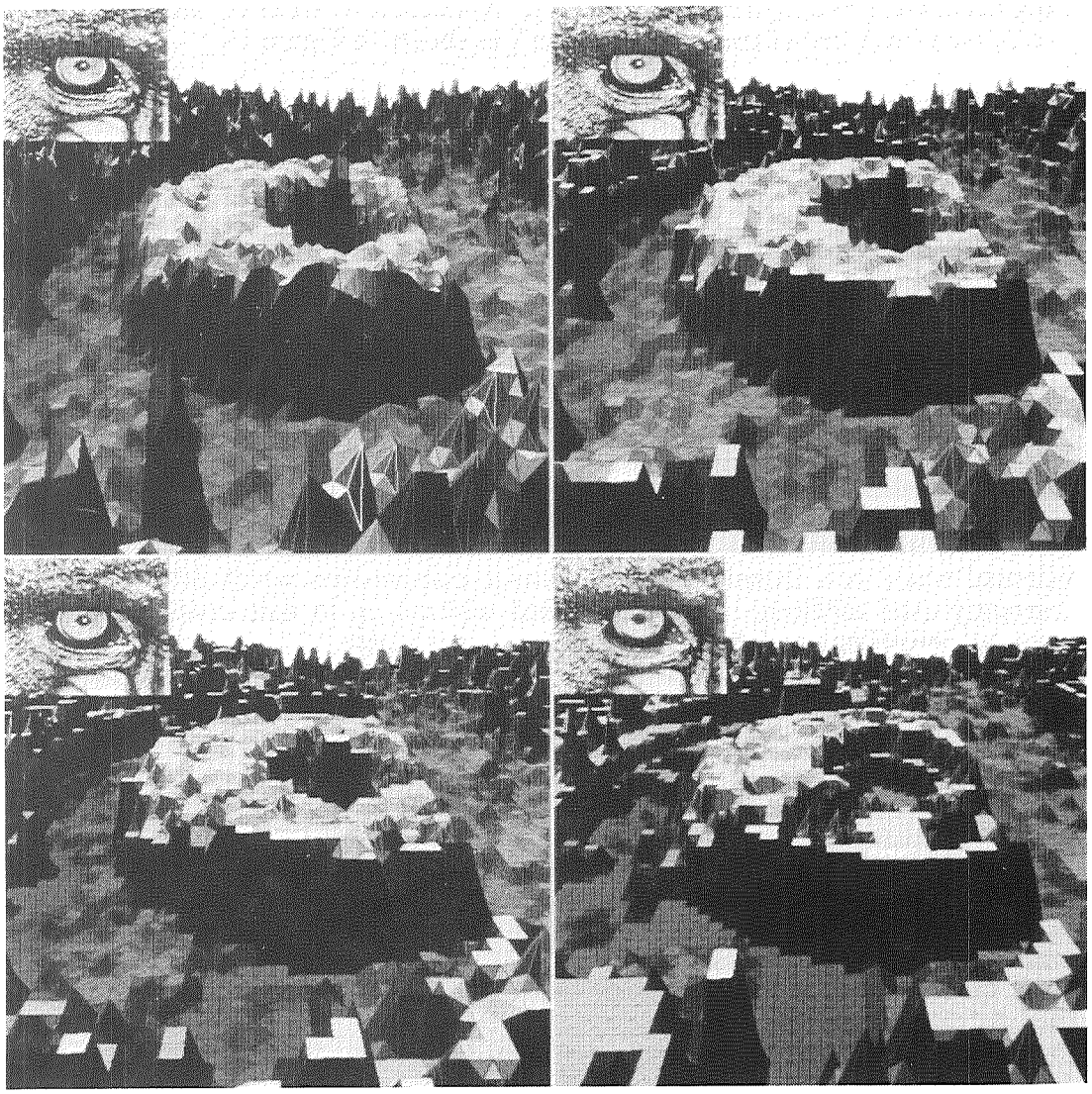


Figure 9. Shown is a sequence of 3D representations of an image which demonstrates how increased threshold values affect the Contour Merging process. The 3D view shows the contours around the mandrill's right eye as seen when standing above the bridge of the nose.

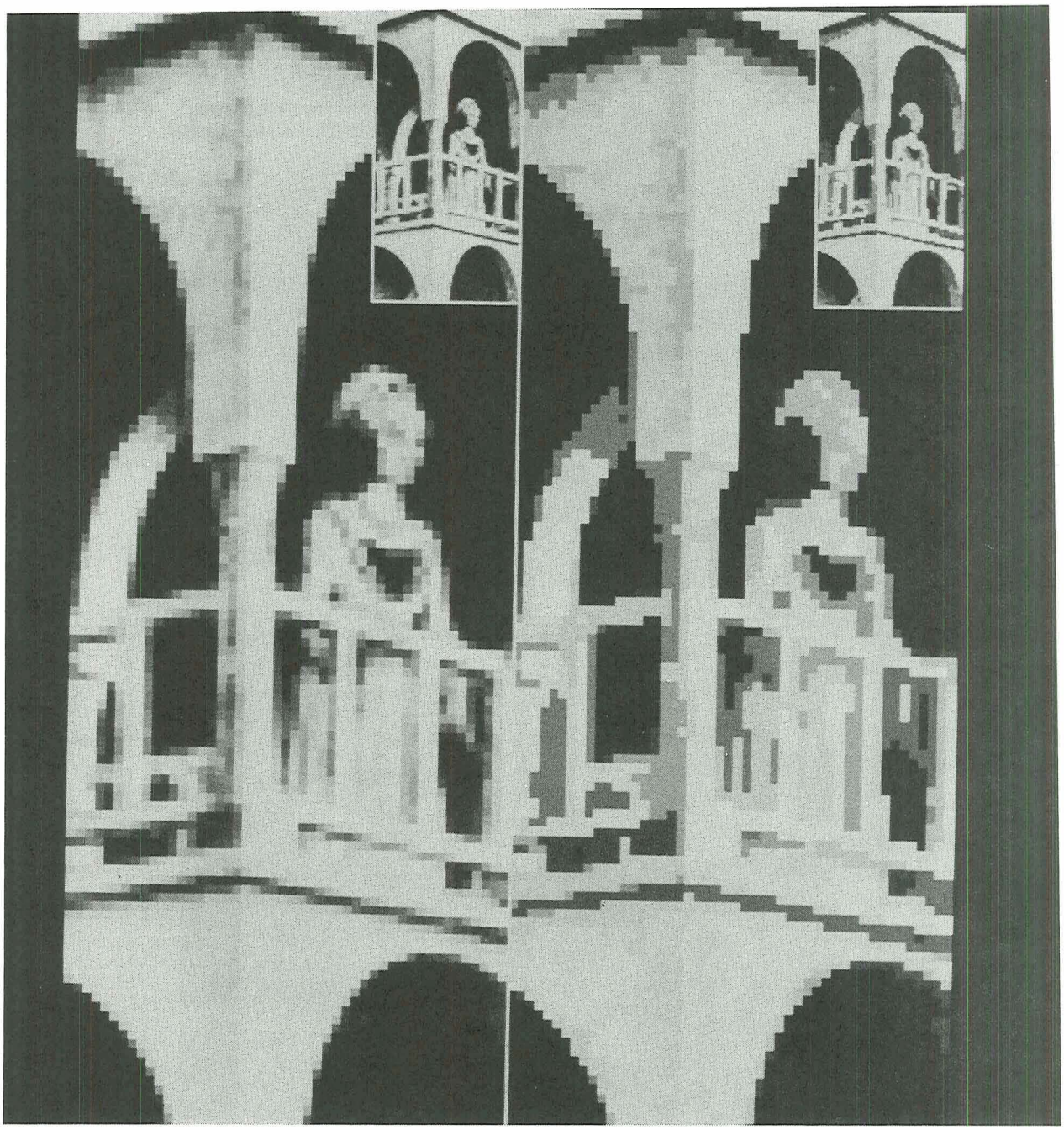


Figure 10. Shown is a blown-up section of the image "eshcer." The image on the right has been coded using the contour merging algorithm. Visual artifacts includes loss of connectivity, slight false contouring, and loss of highlight.

Figure 11. Shown is a blown-up section of the image "garden." The flower bed becomes blotchy large threshold value.

COMPRESSION OF SPECTRAL METEOROLOGICAL IMAGERY

Kristo Miettinen
General Electric, Astro–Space Division
Valley Forge, Pennsylvania

Abstract. Data compression is essential to current low–earth–orbit spectral sensors with global coverage, *e.g.* meteorological sensors. Such sensors routinely produce in excess of 30 Gb of data per orbit (over 4 Mb/s for about 110 min.) while typically limited to less than 10 Gb of downlink capacity per orbit (15 minutes at 10 Mb/s). Astro–Space Division develops spaceborne compression systems for compression ratios from as little as three to as much as twenty–to–one for high–fidelity reconstructions. Current hardware production and development at Astro–Space Division focuses on discrete cosine transform (DCT) systems implemented with the GE PFFT chip, a 32x32 2D–DCT engine. Spectral relations in the data are exploited through block mean extraction followed by orthonormal transformation. The transformation produces blocks with spatial correlation that are suitable for further compression with any block–oriented spatial compression system, *e.g.* Astro–Space Division's Laplacian modeler and analytic encoder of DCT coefficients.

1.0 Introduction

Image compression typically consists of two steps: decorrelation and encoding. For spectral imagery a flexible and practical approach to compression is to transform the *n*–band spectral data into *n* spatially correlated but spectrally mostly decorrelated channels, using some approximation to the spectral Karhunen–Loève transform (KLT). Then the channels can be compressed using any spatial compression technique. The DCT is probably the most effective practical spatial decorrelator for image compression, but it produces a data stream which is difficult to encode due to its severe cyclical nonstationarity[1]. The analytic encoder was developed specifically for such data sources, and has therefore first been applied to DCT coefficient encoding.

The spatial compression system consisting of the DCT, Laplacian modeler and analytic encoder is specifically designed for superior performance at bitrates near those achievable by optimized lossless compressors, producing nearly lossless performance (85% of pixels reconstructed exactly) at the lossless compressor 's bitrate. The DCT–Laplacian system generally outperforms JPEG draft rev. 8 by about 0.5 b/p (or an MSE ratio of about 1:2) when compressing more lossily .

2.0 Spectral Decorrelation

One of the results of our investigations into spectral data compression (a.k.a. "multispectral compression" [sic]) is that data with extension in a spectral dimension can be effectively compressed by first applying a decorrelating orthonormal transformation to each pixel spectrally and then compressing blocks of like coefficients from adjacent pixels spatially as though they were image data.

This is not a new idea, but empirical investigations have demonstrated that three–dimensional (one spectral and two spatial dimensions) optimal decorrelating transforms (Karhunen–Loève or Hotelling transforms) are decomposable into separate spectral and spatial transforms. Thus, there is nothing substantial to be gained by joint spatial–spectral transformations. Whereas such transformations have previously been rejected *a priori* for practical reasons, we now reject them *a posteriori*.

1. The expected moments of transform coefficients vary within blocks, hence nonstationarity, but the pattern is repeated for each block, hence cyclical.

Thus, all that is needed for spectral compression is a decorrelating orthonormal spectral transformation. Once the spectral transformation is done, the resulting channels of coefficients can be compressed using any normal spatial compression technique.

The implementation of the spectral transform is still not simple. The basis vectors of spectral Karhunen–Loève transforms derived for several 5–band AVHRR images varied greatly from scene to scene, and attempting to use a "composite" transform derived from the pooled covariance matrix in a prototype spectral compressor lead to insignificant extra compression (less than 0.1 bits per band per pixel) over a benchmark scheme of compressing each band spatially only, with bits allocated among the bands according to information content.

2.1 Stabilizing the Karhunen–Loève Transform

One basic feature of spectral images which can be overlooked in spatial applications but becomes important in spectral applications is that different surfaces (*e.g.* cloud, water, ice, forest, desert) have different offsets between spectral bands *but similarly aligned random variations about those offsets*. Therefore subscenes with uniform underlying surfaces have similar optimum spectral decorrelating transforms regardless of the surfaces, but scenes with a variety of surface types have optimal spectral transforms which depend on the mix of surfaces, since the intersurface spectral variations swamp the smaller intrasurface variations.

This is graphically illustrated for a two–dimensional system in figure 1 below.

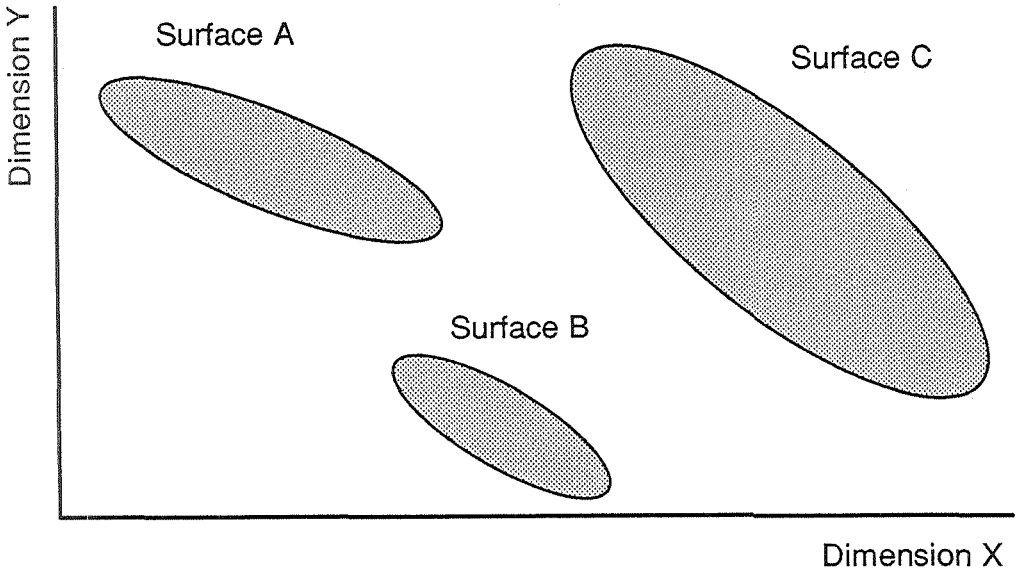


Figure 1 : Example of Alignment of Intrasurface Variations

Clearly in figure 1 the principal component of variation in the X–Y domain is similar for scenes consisting of only one surface type (the main axes of each ellipse are nearly parallel), but for any pair of surface types the principal component of variation in the X–Y domain will be more or less parallel to the vector connecting the centroids of each surface's distribution, while for a scene containing all three surface types the principal component of variation could be oriented in any manner.

Without actually trying to classify pixels and compress only within irregularly shaped regions, the easiest way to eliminate intersurface variations from data being spectrally transformed is to apply

spectral transformation in the spatial frequency domain only on those elements whose wavelength is shorter than plausible spatial feature sizes, *i.e.* those coefficients which register intrasurface variations better than intersurface variations. For DCT compression with small blocksizes (or vector quantization techniques) this may mean spectrally transforming all but the principal spatial components (the "DC terms").

2.2 The Spectral Compressor Front End

A simple and generic implementation of spectral compression as outlined above consists of a standard spatial compressor (*e.g.* the analytic encoder with Laplacian coefficient modeler applied to DCT coefficients) with a centroid extractor and deviate transformer appended to the front end. Assuming *n*–dimensional spectral data and a spatial compressor that operates on 32 by 32 pixel blocks (16x16 and 32x32 are common blocksizes for the DCT, but 4x4 and 8x8 are more common for vector quantization), the front end is depicted in figure 2 below.

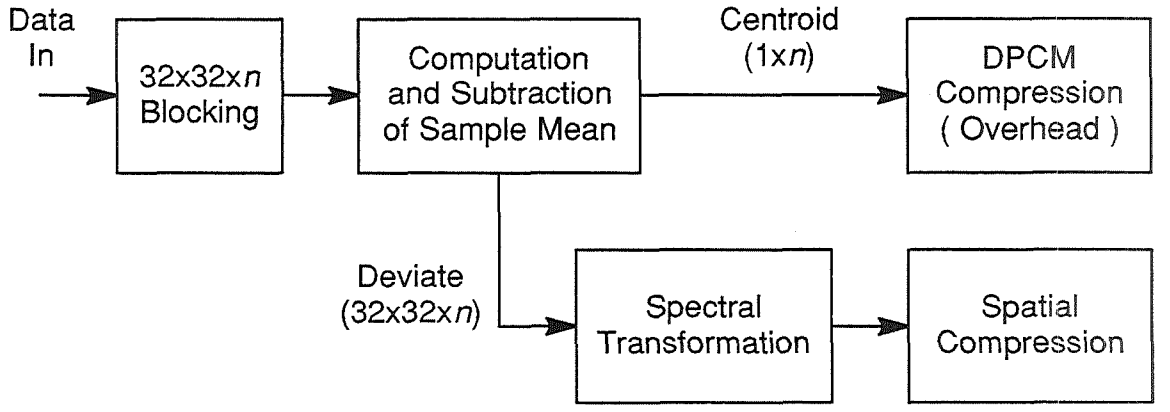


Figure 2 : Front End Block Diagram

When the spatial compressor is a discrete cosine transform–based algorithm the centroid values can be substituted for the DC coefficients of the DCT blocks in the overhead stream, since those coefficients will all be zero for blocks whose centroids have been extracted. Other spatial compressors (*e.g.* mean–residual vector quantizers) also exploit the zero–mean property of the deviates.

3.0 The Spatial Compressor

For the Astro–Space DCT compressor the sensor data are partitioned into 32–pixel square blocks[2], which are transformed into 32–point square blocks of transform coefficients. These blocks are then passed through the Laplacian (analytic) encoder one at a time, with a new arithmetic codeword generated for each block. In between the transform operator and the encoder the data are statistically characterized by a coefficient modeler so that scale predictions can be generated for the Laplacian

2. Previous studies for Mars Rover Sample and Return data compression indicated that the most efective power–of–two sized transforms were 16 pixels and 32 pixels square. The choice of 32 pixels square was based in part on the size of GE's proprietary PFFT chip (1024–point transforms, or 32 transforms in parallel with 32 points each) and in part on the desire to lower the amount of overhead information (which is typically generated on a per–block basis and compressed less efficiently than other data).

encoder. The Laplacian model is purely an empirical choice[3]. This process is depicted below in figure 3.

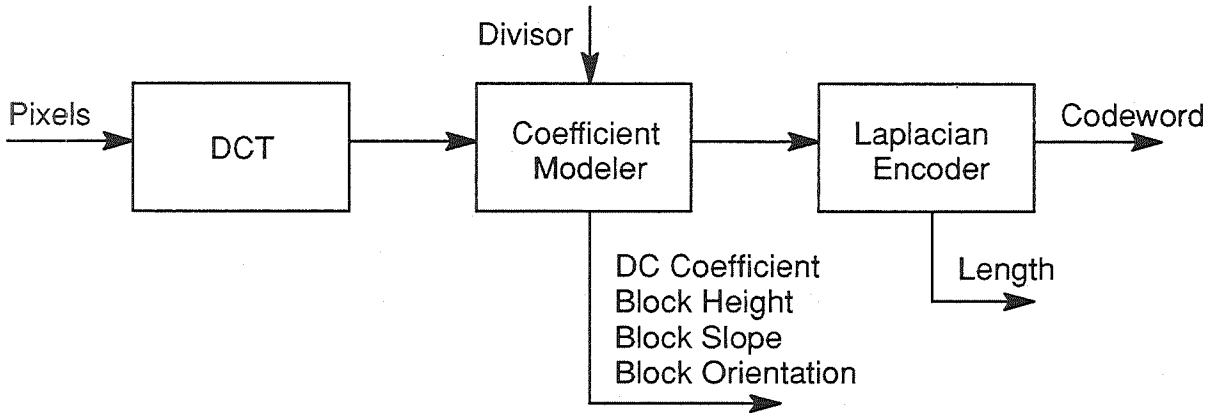


Figure 3 : System Block Diagram

The flow sizes in figure 3 are as follows:

- 1024 pixels enter the discrete cosine transformer,
- 1024 coefficients and one divisor (W_{∞}) enter the coefficient modeler,
- The principal ("DC") coefficient and block model parameters are put out as block overhead for further compression and separate transmission,
- 1023 coefficients (the "AC" coefficients) and associated predictions enter the Laplacian encoder,
- The Laplacian codeword (a long bitstream) and its length are put out, the former as the principal output, the latter as block overhead for further compression and separate transmission.

3.1 The Discrete Cosine Transformer

The Astro–Space discrete cosine transformer takes a 32–pixel square block of data and performs two passes of 32 parallel 32–point one–dimensional discrete cosine transforms. In between the two passes the data are transposed row for column so that the two passes are first horizontally across the block, then vertically across it.

The discrete cosine transform has the property that for typical image data, the transform coefficients are nearly uncorrelated. Specifically, the principal components of samples from first–order Markov processes approach the basis vectors of the discrete cosine transform as the adjacent element correlation coefficient approaches unity. The two–dimensional Markov model is intuitively appealing as an interpretation of images, and typical images have adjacent pixel correlation coefficients in excess of 0.9, so the discrete cosine transform is widely used.

3.2 Decorrelation Efficiency of the Two–Dimensional Discrete Cosine Transform

The covariance hypercube of the coefficients of a two–dimensional transform is simply the four–dimensional transform of the covariance hypercube of the original two–dimensional data block. Given

3. See section 4.4 of Clarke, "Transform Coding of Images", Academic Press, London 1985, for a thorough survey of coefficient distribution models.

an optimal decorrelator (the KLT) the transform domain covariance matrix of one–dimensional data would be diagonal, and the transform domain covariance hypercube of two–dimensional data would be diago–planar (have nonzero entries only on the "diagonal" plane). To quantify the extent to which a transform is optimal, we define two measures of optimality: decorrelation efficiency (the traditional measure) is the proportional reduction in off–diagonal energy (sum of absolute values of covariances) from the data domain to the transform domain[4], and relative remaining energy is the ratio of the total energy in the transform domain to the (planar) trace of the data domain covariance hypercube. Ideally both measures would be one, with decorrelation efficiency approaching from below and relative remaining energy approaching from above.

Figure 4 shows scatter plots of decorrelation efficiency and relative remaining energy as a function of adjacent–pixel correlation[5]. In interpreting these plots it is important to keep in mind that the "optimum" decorrelator corresponding to complete efficiency and no extra remaining energy is only optimum within the following constraints:

- Decorrelation shall only be accomplished within blocks; interblock correlation is not considered,
- Decorrelation shall be accomplished by linear transformation, *and the same transformation must be used on all blocks.*

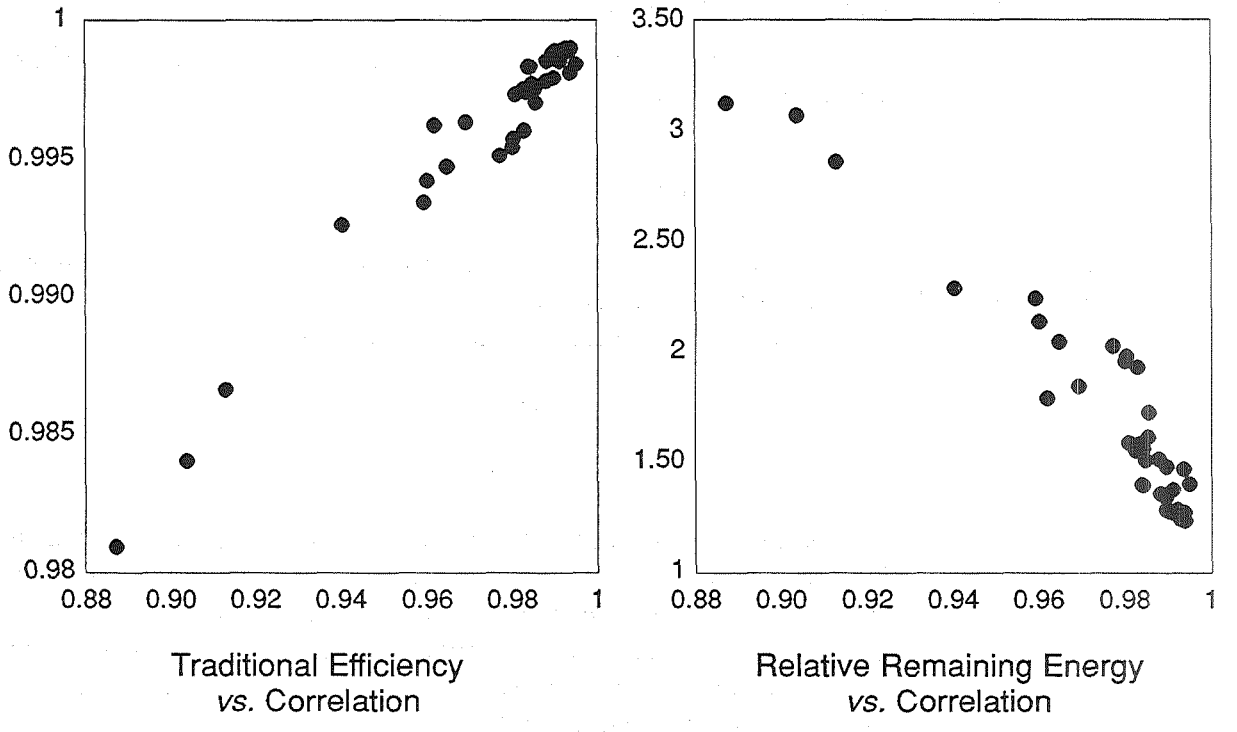


Figure 4 : Decorrelation Efficiency *vs.* Adjacent Element Correlation

4. Diagonal energy is unchanged since: 1) the trace of a matrix is invariant under orthonormal transformation; 2) the covariance matrix diagonal elements are all variances, hence positive; 3) the covariance matrix is real symmetric, hence the transform–domain diagonal elements are all positive.

5. Actually, the independent variable is diagonal correlation with pixels one row up or down *and* one column left or right, raised to the root–one–half power. This is sort of a geometric mean of horizontal and vertical adjacent–pixel correlations, which are not equal for real sensors.

The plots in figures 4 and 5 are based on 35 frames (7 scenes with 5 bands each) of AVHRR data from the 1990 GE benchmark satellite image test suite. The frames are 512x512x10 bits, so efficiency was computed for 16x16 transforms since a 512–pixel square frame does not have sufficient degrees of freedom for a 32^4–element covariance hypercube of full rank.

As the figures suggest, decorrelation efficiency and relative remaining energy are directly related. In the limit, if we assume that all correlations within the 16 by 16 pixel data block are near unity, then the 16^4–element covariance hypercube has 16^2 units of information energy out of 16^4 total units of energy. Expressing relative remaining energy in terms of decorrelation efficiency η (the portion of the 16^4–16^2 units of redundant energy removed),

$$\frac{E_r}{E_o} \approx \frac{16^2 + (1-\eta)\left(16^4 - 16^2\right)}{16^2} = 256 - 255\eta.$$

This approximate relationship is apparent in figures 4 and 5. Clearly relative remaining energy is more sensitive than the traditional measure of efficiency. Note that changing the size of the block and the assumed amount of non–diagonal energy in the covariance hypercube changes the two coefficients in the right–hand expression above without affecting their unit difference. For 32–pixel square blocks the coefficients in the expression are about one thousand.

Figure 5 indicates a typical relative remaining energy of two for the test suite. Using the relationship of entropy (ϕ) as a function of energy (or variance, σ^2)

$$\phi_\sigma \approx \log_4 \sigma^2 + 2$$

a relative remaining energy of two means a half a bit per coefficient (or pixel) of extra information is needed to encode the not–quite–decorrelated data set. Using the relationship between lost resolution b (bits per symbol) and mean squared error for densely quantized data

$$MSE = \frac{2^{2b} - 1}{12}$$

a half of a bit of resolution lost causes a mean squared error of $1/12$ at a bitrate that would otherwise have been lossless, or doubles the mean squared error at other bitrates. While the discrete cosine transform is good, it is still not nearly optimal even within the constraints given above for this efficiency analysis. For this reason Astro–Space division continues to study DCT pre– or postprocessing, *e.g.* sparse matrix pre– or postmultiplication to exploit the DCT as a first (or last) easily computable step on the way to a good approximation of the KLT.

3.3 Coefficient Modeling and Quantization

The coefficient modeler, quantizer, and analytic encoder are optimized for data having the Laplacian distribution

$$f_x(x) = \frac{1}{2s} e^{-|x|/s}.$$

The coefficient modeler predicts the value of |x| from block overhead parameters indicating general block energy, energy decay with sequency, and energy orientation, and from coefficient reconstructions that will be available to the decompressor at the time it is working to decode any particular coefficient. The prediction of |x| for any coefficient is used as s (the dispersion parameter of the Laplacian distribution) for quantizing and encoding that coefficient. Clearly, s is not constrained to be constant for any block of coefficients, nor even to vary slowly.

The quantizer design is based upon the following premises:

- A tradeoff ratio $-R$ is set for expected squared error σ^2 *vs.* expected code length λ,
- The center quantizer bin will be symmetric about zero,
- The center bin will be set to that width (W_0) for which the marginal increase in σ^2 for expanding that bin at each end is R times the marginal reduction in λ,
- Subsequent bins will extend outward toward positive and negative infinity from the endpoints of previous bins until the marginal increase in σ^2 for expanding those bins is R times the marginal reduction in λ, at which point those bins will end (at width W_0) and new bins will begin.

Varying R in this scheme will sweep through the various points on the limiting curve of output rate *vs.* distortion, with each value of R producing that point on the rate–distortion curve where the derivative of distortion with respect to rate is $-R$.

Solving the equation

$$\frac{\partial \sigma^2}{\partial \lambda} = -R$$

for the upper bound b of a quantization bin as a function of the lower bound a, the cost constraint R, and the dispersion s of the laplacian–distributed variable x, and also solving for a symmetrical bin about zero (for which $a = -b$) yields quantizers with uniform size bins of f zero, a slightly larger bin about zero, bin sizes that increase slightly with s, and everything varying in the same proportions when R is varied. In particular, if we define W_∞ to be the width of a quantizer bin for the uniform distribution given cost constraint R, then the bin widths and reconstruction points (constrained expected values of positive x given $a \le x \le b$, where a and b are the bin boundaries for positive nonzero bins) are tabulated below in proportion to W_∞ for some values of s.

From Table 1 it is apparent that the quantizer favors finer quantization of low–energy signals (high–sequency DCT coefficients) than high–energy signals, since distortion reduction is "cheaper" for them in proportion to output expansion.

Such a quantization scheme obviously fixes neither block distortion nor block output rate. High–energy blocks will obviously consume more bits than low–energy blocks given the same value for W_∞ (the parameter used to control overall compression quality, maintained constant for long periods of time), but they will not consume enough extra bits to match the output quality of the low–energy blocks because marginal distortion improvements are more expensive for high–energy signals.

Dispersion of Distribution	Width of Bin About Zero	Width of Nonzero Bins	Reconstruction Point
s	W_0/W_∞	W_0/W_∞	$E(x-a)/W_\infty$
0	$2/\sqrt{6}$	$1/\sqrt{6}$	0
0.01	0.81699	0.41837	0.01000
0.02	0.81845	0.42874	0.02000
0.05	0.82852	0.46123	0.04995
0.1	0.85530	0.51548	0.09701
0.2	0.89775	0.60094	0.16867
0.5	0.94784	0.73621	0.28087
1	0.97151	0.82913	0.35792
2	0.98509	0.89898	0.41593
5	0.99387	0.95441	0.46203
∞	1	1	$1/2$

Table 1 : Quantizer Parameters

The actual quantizer is simply realized as two interpolated lookup tables: one yielding the inverse of the nonzero bin size and the other yielding half the difference between the zero and nonzero bin sizes. Using these values the coefficient x is quantized into the integer i according to

$$i = \frac{x \pm \frac{W_0 - W_0}{2}}{W_0}$$

with i rounded to the nearest integer, and the alternating sign matching the sign of x. The formula for i above is designed to map the quantization bin boundaries a and b to odd multiples of $1/2$, so that all values of x in the same bin round to the same i.

3.4 The Laplacian (Analytic) Encoder

The analytic encoder is an arithmetic encoder which uses an analytic expression for the c.d.f. of the distribution that a particular datum is assumed to have been drawn from, rather than a preprogrammed table or prior experience. Thus, along with each datum the encoder expects as many parameters as are needed to define the c.d.f. In the case of the laplacian distribution, the only parameter needed is s, the expected absolute value.

The Laplacian encoder is a state machine which takes in Laplacian–distributed data coupled with scale predictions and puts out a bit stream called a codeword. The bit stream is unlimited in length, and is restarted for each 32–pixel square block in Astro–Space division's current implementation of DCT compression. The Laplacian encoder can terminate a codeword in either of two ways: simply

by putting out (as compressible overhead) the length of the codeword, or by extending the codeword (theoretically by $2\frac{1}{3}$ bits on average) so that regardless of what follows the codeword will be properly decoded. The former option is preferable for applications where data losses are possible, as it allows a total restart at the beginning of each block.

Freedom to encode diverse data in any order, rather than in statistically homogeneous groups, brings a number of benefits:

- Data can be grouped spatially, for instance encoding all of the coefficients of a single discrete cosine transform block as one symbol set. This obviously simplifies buffering and also provides robustness. For instance, if a 32 by 32 pixel block is compressed to one bit per pixel, then one thousand consecutive bits will all be from the same block, so that burst errors in communication are likely to affect only one block at a time (or two at most).
- A carefully designed predictor of |x| used to choose the distribution dispersion parameter s will be able to adjust immediately to any nonstationarity in the data source, tracking variations in source statistics on the single–sample level rather than from sample set to sample set.
- No symbol distribution or similar overhead needs to be transmitted.

The codeword put out by the Laplacian encoder is the principal compression product for each block. For instance, for a compressor operating at four bits per pixel the codeword would be, on average, about four thousand bits for each 32x32 block (though unusual blocks could yield much longer or much shorter codewords for the same setting of the control parameter W_∞).

3.4.1 Encoder Operation

The Laplacian encoder differs from normal arithmetic coding by not grouping similar data together: each datum is encoded individually as though it came from a large sample, but each datum is in fact a sample of one. Instead of having a probability table the Laplacian encoder has embedded in it the analytic expression for the cumulative density function of a Laplacian distribution,

$$F_x(x) = \frac{1}{2}e^{x/s} \quad | \; x \le 0, \qquad F_x(x) = 1 - \frac{1}{2}e^{-x/s} \quad | \; x > 0.$$

Each datum is modelled as coming from a Laplacian distribution with mean absolute value s, with a unique value of s for each datum. Obviously, s cannot be a measured sample statistic (there is only one datum in the sample); it must be a prediction entirely independent of the datum to be encoded. As long as the predictor is unbiased and has the power to discriminate, i.e. as long as

$$E(x)|s = s$$

while

$$0 \ll \sigma_s < \mu_s$$

the encoder will operate as efficiently as, and often more efficiently than, encoders which rely on grouping similar data together.

3.5 Implementation of Laplacian Encoder/Decoder

The encoder follows the general procedure for arithmetic encoders, maintaining a bounded interval within which the ultimate codeword will fall, and transmitting bits (and shifting up the boundaries) as the boundaries converge to the point where leading bits match. Specifically, a pass through the encoder (with one symbol) consists of the following:

(1) Computation of Hi and Lo, the boundaries of the interval on [0, 1] where the codeword will be. The computation yields a subinterval on the stored previous interval Hi– and Lo+ (derived by truncating the previous Hi and Lo inward to some buffer size, e.g. 16 bits) corresponding to the c.d.f. interval between the bin boundaries a and b for the input datum i and the dispersion s. The new Hi and Lo are temporarily maintained at their full precision, e.g. 32 bits if Hi–, Lo+, and the c.d.f. points are all 16–bit numbers.

(2) Transmission of as many of the leading bits as match in Hi and Lo (actually, for exception handling and sneaky storage purposes, transmission of "0" if Hi $\leq \frac{1}{2}$ and "1" if Lo $\geq \frac{1}{2}$, repeated as often as possible). As the bits are transmitted, Hi and Lo are shifted up (padding with zeroes) so that in the end

$$0 \leq \text{Lo} < \frac{1}{2} < \text{Hi} \leq 1.$$

(3) Truncation of Hi and Lo into new values of Hi– and Lo+. As the labels imply, Hi– is derived by rounding Hi down while Lo+ is derived by rounding Lo up. This rounding towards the inside of the interval guarantees that adjacent intervals do not overlap after truncation. If they did, a codeword in the overlapping region could be misinterpreted.

Mathematically the decoding process for one coefficient consists merely of solving the actual codeword, Hi, and Lo, for a c.d.f. value, and then evaluating s and the inverse of the Laplacian c.d.f. for a value of x', which is quantized to i (note that the actual value of x' produced by the inverse c.d.f will lie in the same quantization bin as the original value of x, but beyond that it carries no information, and therefore should be reduced to its bin label i). The true value of x will then be estimated by adding (subtracting for negative i) a tabulated and interpolated expected value (the last column in table 1 above) to the quantization bin boundary a (or $-a$ for negative i).

4.0 Block Overhead Compression

For each block of 1024 pixels the compressor puts out four pieces of overhead information that need to be transmitted along with the codeword for reconstruction:

- The raw DC coefficients (a 5–dimensional quantity),
- The codeword length (in bits),
- The "height" of the scaled, quantized coefficient block absolute values (a 5–dimensional quantity),
- The "slope" of the scaled, quantized coefficient block absolute values (a 5–dimensional quantity).

- The "orientation" of the scaled, quantized coefficient block absolute values (a 5–dimensional quantity).

The "height" of the coefficient block is a measure of information content and therefore obviously correlated with the codeword length, and the two are also be weakly correlated with the "slope" of the coefficient block. The DC coefficients, codeword lengths and coefficient block "heights" are also obviously correlated with their counterparts in adjacent blocks, and the coefficient block "slopes" are also weakly spatially correlated. Finally, all of the 5–dimensional quantities have internal correlation among their scalar components. These correlations are exploited in a DPCM compressor of overhead information. The codeword length is compressed losslessly, and the other quantities are compressed lossily, with the compressor using those values of lossily compressed parameters that the reconstructor will have rather than the original values.

5.0 Performance of the Compressor on Meteorological Images

Astro–Space division maintains ongoing efforts to determine the value to end users of lossily compressed imagery. Studies have included generic quality measures (*e.g.* mean squared and absolute errors, frequency distortion, automated pattern recognition, edge detection and location, difference histograms and images, *etc.*), user–oriented objective quality measures (*e.g.* calculated sea surface temperatures, spatial coherence analyses, cloud type identification, cloud cover calculation, forest classification, crop identification, ice fissure detection, location, and measurement, interplanetary probe landing site hazard identification, surface altitude and grade derivation from stereo pairs, *etc.*), and subjective expert assessments. Algorithms have been evaluated not only for the quality of their reconstructed images but also for hardware requirements at the transmitter, vulnerability to communications errors, compatibility with packeting and encryption, and exploitation of error–correcting codes and retransmission.

Most analyses have indicated a surprising robustness of image–derived information to compression losses, particularly to compression errors without structure, *e.g.* white noise. A battery of meteorological tests on 5–band AVHRR data indicated that with our best compression techniques compression ratios of 50:1 were achievable with borderline unacceptable defects in the images. Images compressed 25:1 were clearly acceptable for all analyses, and images compressed between 5:1 and 10:1 exhibited either no loss (due to rounding to some user–selected relevant resolution) or negligible loss in image–derived quantities.

For example, one study focussed on computing multichannel sea surface temperatures (MCSST) from compressed AVHRR data. Table 2 below summarizes root–mean–square (RMS) errors in degrees Kelvin for a variety of compressors.

Compression Ratio	DCT/Laplacian	JPEG Draft Rev. 8	DPCM	MRVQ
5:1	0.146 °K	0.292 °K	0.388 °K	0.331 °K
10:1	0.289 °K	0.588 °K	0.863 °K	0.560 °K
20:1	0.396 °K	0.811 °K	1.233 °K	0.789 °K

Table 2 : Sea Surface Temperature RMS Errors

The data in table 2 were interpolated for those compressors (all but DCT/Laplacian, which employs rate feedback control) which don't produce an arbitrary compression ratio exactly on demand. The

DCT Laplacian algorithm used involved no spectral compression or spectral resource allocation in order to produce results commensurate with the other (purely spatial) techniques.

The RMS errors of figure 2 were computed on a pixel–by–pixel basis, and would seem (in some cases) to be unacceptable, except that typical sea surface temperature analyses involve computing average temperatures over regions of not less than 256 pixels. Such averaging reduces rms error by factors in excess of 16, making even the worst compressed data look good. Similar averaging effects help in other meteorological analyses, *e.g.* cloud cover calculations, since most analyses focus on the properties of large objects (*e.g.* cloud formations, ocean currents, polar icecaps, *etc.*).

Another interesting result of our user–oriented studies is that scalar error measures are usually entirely adequate for intercomparisons of conceptually dissimilar algorithms. For instance, when we presented a panel of meteorologists with a suite of images compressed to various degrees with various algorithms and asked them to subjectively grade the reconstructions, the rank order in which they placed the reconstructions exactly matched the rank order of their mean squared errors (MSE), without regard to whether the images were compressed using any of a number of DCT algorithms, a DPCM algorithm, or an MRVQ algorithm, all of which produced different characteristic errors. In the spirit of this observation in defense of the much–maligned MSE, figure 5 below presents the MSE performance of the DCT/Laplacian compressor, both with the spectral transform and without, for a particularly challenging scene combining clouds on two levels, land, open water, and polar ice.

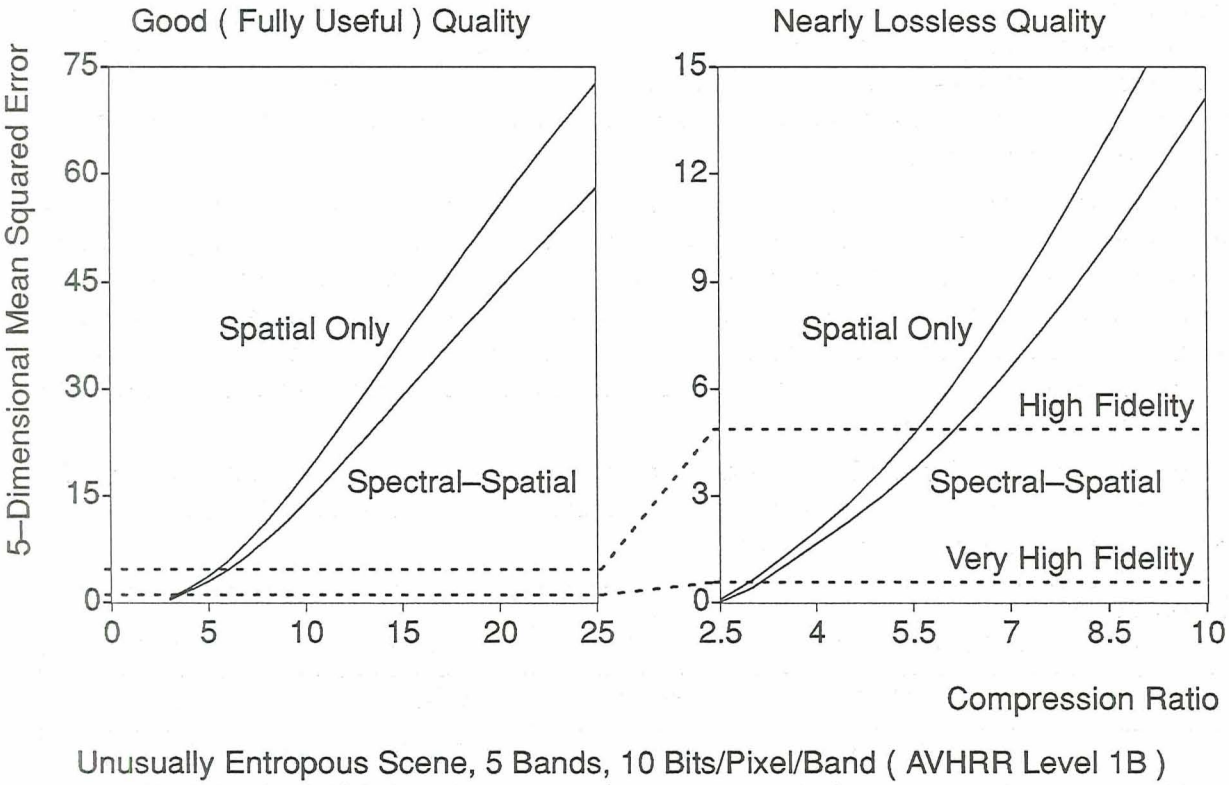


Figure 5 : Bad–Case Compressor Performance

Operating the spectral compressor without the spectral transform but with the same value of W_∞ used for each spectral band causes the same cost constraint R to be used across the whole image, resulting

in bit allocation among the bands as well as among the blocks in such a manner as to reduce overall distortion as much as possible within the tradeoff ratio $-R$ of distortion for output. The difference in performance shown in figure 5 is due entirely to spectral decorrelation, and can be used as a measure of the added benefit to be gained from full spectral/spatial compression over spatial compression with output bits allocated among the spectral bands according to need. The latter approach suggests itself as a way to compartmentalize the effects of single–band sensor hardware failures such as that on the AVHRR sensor aboard TIROS–7.

Figure 5 shows two statistical classifications of reconstruction quality in use at Astro–Space Division: High Fidelity reconstruction is defined as reconstruction with an MSE of one squared quantum per scalar dimension (chosen for parsimony and uniphilia), and Very High Fidelity reconstruction is defined as reconstruction with an MSE of $1/12$ of one squared quantum per scalar dimension (chosen to match quantization error for dense quantizers). Together with Lossless and Lossy these provide a range of qualitative descriptors of reconstructions.

6.0 Choosing Spectral Bands for a Sensor

Existing sensors designed without spectral data compression in mind (such as AVHRR) tend to have spectral bands which are relatively uncorrelated. This is only natural, since without spectral compression the cost (in storage space and transmission bandwidth) of adding a new band to a sensor is independent of the added nonredundant information in the band, so bands are chosen to provide as much nonredundant information as possible. Intuitively, when a sensor is designed without spectral compression in mind, the optimal sensor design produces spectrally irreducible data, thereby packing in the most information per band, and justifying the omission of spectral compression after the fact. This leads to spectral compressor performance such as that shown in figure 5, where exploiting spectral redundancies provides only limited performance improvements over well–designed spatial compression.

Adding spectral compression to a sensor changes the cost of adding a new band by exploiting spectral redundancies so that a system designer has only to pay (in bytes, baud, and bucks) for the nonredundant information in a band, rather than paying a fixed price for all the information in a band. Since the cost and added value of a new band will vary together given spectral compression, there will no longer be an obvious incentive to choose highly uncorrelated bands for a sensor. Future sensors with spectral compression will therefore tend to have more spectral bands, and more correlation among bands, than current sensors designed for the same purposes.

7.0 Choosing Sensor Output Quantization Resolution

In parallel with our user–oriented studies Astro–Space division also conducts systems studies to assess the impact of data compression on sensor design, backplane capacity, buffer memory requirements, and downlink encryption and error correction. In this context, one of the most common customer requirements is for lossless data compression regardless of other systemic, dynamic, and random sources of loss. Astro–Space Division develops lossless compressors for various uses, but future sensors ought to be designed for lossy compression, since lossy data compression applied to higher–resolution data always produces better radiometric accuracy for a given bitrate than lossless data compression applied to lower–resolution data.

This is intuitively a consequence of the fact that the underlying "exact" data values are real numbers, and the difference between higher–resolution sensor data and lower–resolution sensor data is simply the degree of truncation, with the lower–resolution data always worse than, and reproducible from,

the higher–resolution data (higher–resolution data is produced by a quantizer with bin boundaries at all of the boundaries of the lower–resolution quantizer, plus other intermediate boundaries as well). Since the lower–resolution data is reproducible from the higher–resolution data, the choice between lossless compression of the lower–resolution data and lossy compression of the higher–resolution data becomes a choice between two lossy approaches to compressing the higher–resolution data: truncation followed by lossless compression or direct lossless compression. Direct lossless compression has flexibility to incur losses in some optimal manner, and therefore should always outperform truncation followed by lossless compression, in which the losses are fixed in an arbitrary manner.

Retaining higher–resolution data for lossy compression is rarely a matter of buying a more precise (and hence more expensive) analog–to–digital converter (ADC), since most sensor data needs to be calibrated, corrected for nonlinear sensor response, spatially resampled, *etc.* The calibrated/corrected/resampled/processed data will be real numbers even though the ADC produces only integers, so as long as some of these steps (and preferably all of them, which is feasible with current spaceborne processors) occur before compression, the data will have to be truncated once more (or many times more) before compression, and these later truncations will determine the resolution of the data being compressed.

The effect of original data resolution on reconstructed data quality can be assessed using the previously presented formula for MSE as a function of bits truncated,

$$MSE = \frac{2^{2b} - 1}{12} .$$

This formula can be inverted to get effective resolution lost as a consequence of error, which, combined with the original resolution, yields effective remaining resolution:

$$R_e \equiv R_0 - \log_4(12MSE + 1) .$$

Effective remaining resolutions can be compared for reconstructions independently of the original data resolution, so one way to assess the value of extra bits of resolution is to measure how many bits of effective resolution are gained by adding a few bits of resolution to the original data, given that in either case the data have to be compressed to the same fixed bitrate before reconstruction.

Such a study is presented in figure 6 below: 10–bit AVHRR data and 8–bit truncations of the same data were compressed to a number of bitrates and reconstructed, and effective resolutions compared. The improvement in effective resolution from having two extra bits in the original data is plotted below as a function of the effective resolution lost in compressing the 8–bit data, so the benefit of extra resolution is expressed as a function of how much of the lesser resolution data are being lost to compression. Clearly, as less of the information in the lower–resolution data is being retained (*i.e.* lossier compression is being performed), the benefits of having even more resolution in the original data evaporate.

Figure 6 shows that substantial gains can be made by increasing original data resolution until the resolution of the data being compressed is such that at least two bits of effective resolution are lost in the compression. Another inference from the figure above is that lossless compression of an image (compressing a lower–resolution image without losing any effective resolution) is about one bit

worse in effective resolution than lossier compression of a higher–resolution version of the same image.

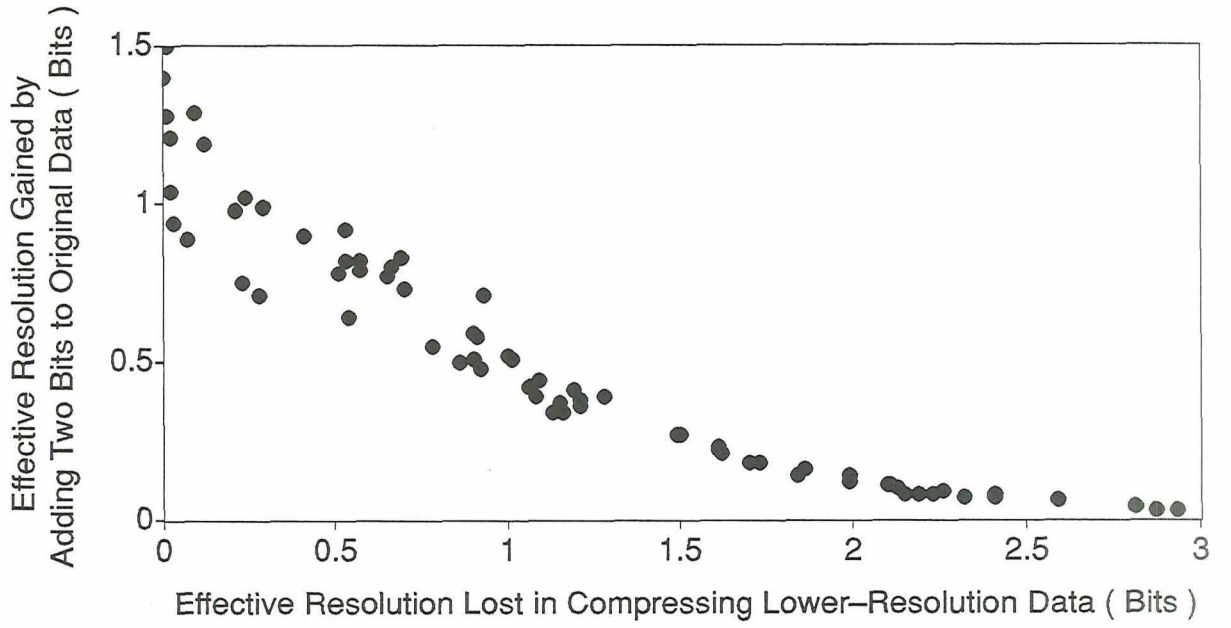


Figure 6 : Benefits of Increasing Original Data Resolution

8.0 Conclusion

Astro–Space Division continues to investigate, develop, and build compression systems for high–volume spectral images. Existing techniques compress data at ratios from 3:1 with pristine reconstruction to 25:1 with acceptable reconstruction quality. Data compression is being designed into future sensors, and future sensors are being designed differently because of data compression. In particular, sensors designed with lossy data compression in mind produce higher resolution calibrated, corrected, and resampled data, and spectral sensors designed with spectral compression in mind can afford more spectral bands since the cost (in storage volume and downlink bandwidth) of adding a new band depends only on the amount of new information in the band.

Methods of Evaluating the Effects of Coding on SAR Data

Melanie Dutkiewicz Ian Cumming
MacDonald Dettwiler and Associates
Richmond, B.C., Canada, V6V 2J3.

March 20, 1992

Abstract

It is recognized that mean square error (MSE) is not a sufficient criterion for determining the acceptability of an image reconstructed from data that has been compressed and decompressed using an encoding algorithm. In the case of Synthetic Aperture Radar (SAR) data, it is also deemed to be insufficient to display the reconstructed image (and perhaps error image) alongside the original and make a (subjective) judgment as to the quality of the reconstructed data. In this paper we suggest a number of additional evaluation criteria which we feel should be included as evaluation metrics in SAR data encoding experiments. These criteria have been specifically chosen to provide a means of ensuring that the important information in the SAR data is preserved. The paper also presents the results of an investigation into the effects of coding on SAR data fidelity when the coding is applied in (a) the signal data domain, and (b) the image domain. An analysis of the results highlights the shortcomings of the MSE criterion, and shows which of the suggested additional criterion have been found to be most important.

1 Motivation for the Study

Since the launch of ERS-1 in July 1991, SAR data has been received in large volumes on a routine basis (approximately 200 GBytes per day). In addition, 4 other SAR satellite sensors are planned for launch in the next decade. Data compression is needed because of the limitations of on-board tape recorders, downlinks, ground archives and data dissemination networks.

SAR data merits study on its own, because its statistical properties are quite different from those of data from other sensors. Most notably, it has a greater high frequency content and a lower signal to noise ratio (SNR) than most other sensors. Thus, compression algorithms which are optimal for other sensors may require modification before the best performance is achieved for SAR data. Also, evaluation metrics which suffice for other data sets may be insufficient for SAR data.

The main distinguishing feature of SAR data is that it is received at the satellite in the form of a hologram (the information governing the spatial distribution of energy in the image is stored in the phase of the received data). This initial received or "signal" data has a frequency spectrum similar

to that of white noise. A numerically intensive convolutional process is required to form the image, *i.e.* convert the signal data into a recognizable scene. The image formation is done on the ground.

SAR signal and SAR image data have quite different statistical properties, particularly with regard to their dynamic range, sample-to-sample correlation, and overall redundancy. For this reason and, for operational reasons, the encoding of SAR data must be studied in two distinct parts, (a) signal data encoding and (b) image data encoding. It is likely that the optimal encoding algorithm will be different in the two cases, and that the encoding performances will be quite different. Also, metrics which are used for analysing the effects of encoding on images may not be suitable for analysing the effects of encoding in the signal data domain.

2 Evaluation Metrics

The evaluation environments for SAR signal and SAR image data are given in Figure 1. As the figure shows, the analysis of SAR signal data encoding has been divided into two parts. First, the effects of encoding in the signal domain will be analysed in the signal data domain itself, by comparing original and reconstructed signal data sets using the metrics listed in Section 2.1. Second, the original and reconstructed signal data sets will each be processed, and the "original" and "reconstructed" images thus obtained will be analysed and compared in the image domain using the metrics given in Section 2.2.

2.1 Signal Domain Metrics

The properties chosen for evaluation in the signal data domain are given below. Note that each measurement is applied to the original and reconstructed data sets and the results are compared, *or* applied directly to the error data set.

1. Data Statistics
   Comparison of data range, mean, standard deviation, kurtosis and entropy of the original and reconstructed data sets may highlight changes in the data characteristics. Analysis of the error data statistics (*e.g.* MSE) will show the characteristics of the error signal.

2. SQNR
   The signal to quantization noise ratio is a measure of the SNR due to encoding. It will be expressed in two ways: the ratio of peak signal to MSE in dB (PSQNR); and the ratio of average signal energy to MSE in dB (ASQNR). MSE gives the total absolute encoding error between original and reconstructed data sets. ASQNR and PSQNR give relative measures useful for comparing results of encoding between data sets with differing mean and peak values respectively.

3. Data Histograms
   Comparison of the histograms of the original and reconstructed data sets will show alterations in the probability distribution brought about by coding. The distribution of the error will be shown in the error data set histogram.

4. Phase Error
   Phase information is used by the processing algorithm to focus the SAR image. A measurement of the phase error present in the reconstructed signal data set will allow comparison of the amount of focusing error introduced by different encoding algorithms.

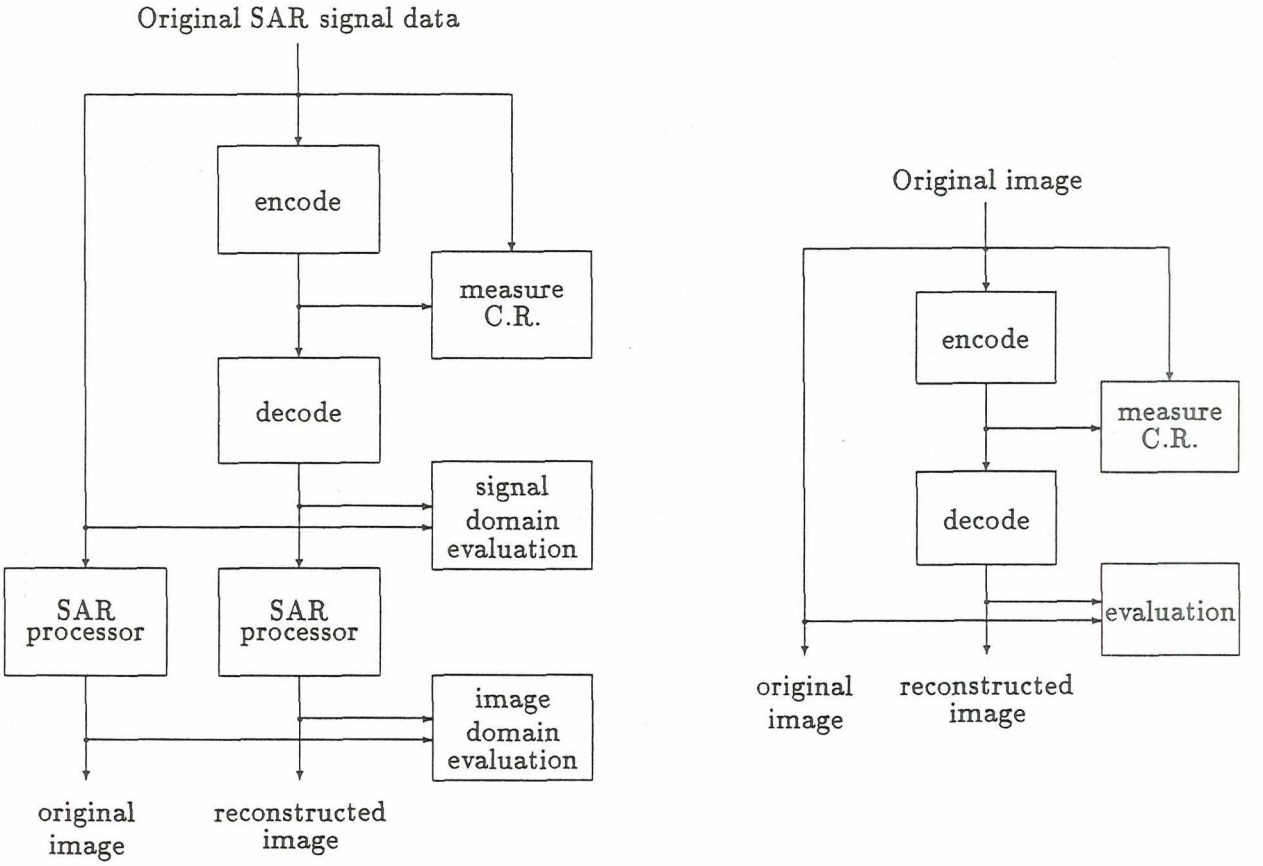


Figure 1: Evaluation environments for SAR signal data encoding (left) and SAR image data encoding (right)

Many of the evaluation metrics which are applicable in the image domain (see Section 2.2) are not applicable in this domain, *e.g.* since the appearance of signal data sets is equivalent to white noise, the effects of added quantization noise on this data will not be discernible, thus no information would be gained by examining the signal data images. These metrics will however be applied in the second part of the signal data analysis, *i.e.* after processing of the original and reconstructed signal data sets.

2.2 Image Domain Metrics

The metrics used in the image domain include items 1, 2 and 3 given in Section 2.1. Item 4 is not relevant in this case, since 4 look *detected* imagery, *i.e. real* data, is assumed. Phase error would however be relevant for single look complex imagery. In addition to the three given metrics, the metrics listed below will be used.

1. Image Appearance
   Comparison of the reconstructed image with the original image will show how well the visual quality of the image is preserved by the coding process. The appearance of the error image will also be analysed for structural content, to see if the error is correlated with any specific feature type in the original image, and for spatial effects such as error propagation.

2. Spectra
   Both the 1D and 2D spectra of the reconstructed image will be compared with that of the original image, and the 1D and 2D spectra of the error image will be examined. The latter should highlight any periodicities in the error, and will also show whether the error is greater for high or for low frequency image components.

3. Image Registration
   Cross correlation between the original and reconstructed data sets, and between small corresponding regions within these data sets, will show if the coding has given rise to any mis-registration effects, either on a global or on a local scale, respectively. The latter could be important in target location and tracking, or in mapping applications, for instance.

4. Image Radiometry
   The changes in the radiometric properties: mean, standard deviation, radiometric resolution and speckle statistics, brought about by coding will be measured. The radiometric linearity of the coding process will also be determined, by plotting the mean values of homogeneous regions in the reconstructed image against the mean values of the same homogeneous regions in the original images.

5. Point Target Characteristics
   The position, resolution and energy distribution of point targets will be computed and compared for the original and reconstructed data sets.

6. Wavelength Estimation
   The magnitude and frequency of the main peaks in the 1D range and azimuth spectra of the original and reconstructed images will be compared.

It is recognized that this list is not exhaustive, however it is felt that it is sufficient to give a detailed insight into not only the magnitude of the error, but the characteristics of the error. Figures relating to these metrics will inform the scientist using the reconstructed SAR data what information may have been altered in the encoding process and to what extent. It is important that any end user of a reconstructed SAR data should have access to such information in order to decide how much trust can be put in conclusions drawn the data.

Additional metrics which could be considered are: edge analysis; and application specific criteria such as effect on segmentation algorithms, or on ship or ice tracking algorithms.

3 Characteristics of Data Used in Study

The data used in this study was data derived from that obtained from the 1978 SEASAT SAR sensor. The major characteristics of this sensor are [Jor80]:

- radar wavelength of 0.235 m,
- incidence angles between 19deg and 25deg,
- nominal altitude of 800 km,
- noise equivalent σ_0 of -21$\pm$5 dB (this implies that the SNR is in the 0 to 10 dB region),
- signal data digitized to 5 bit real samples at an IF of 11.25 MHz,
- resolution of 25 m with 4 looks.

SAR Signal Data

For the signal data encoding studies, the SEASAT data was first downconverted from real samples at the IF of 11.25 MHz to complex samples at baseband, since present day SAR sensors (*e.g.* ERS-1) prefer to record complex baseband data. The components of the complex data are the In-phase (I) and Quadrature (Q) components. Prior to basebanding, the data was scaled to make more use of the 8 bits available after basebanding. Thus the characteristics of the SAR signal data are as follows:

- SEASAT, downconverted to baseband,
- data precision at baseband: 8 bits per I and Q sample,
- data entropy: approx 5 bits,
- data type: signed byte,
- data range: [-128,127],
- data properties:
  - approximately zero-mean, Gaussian distribution,
  - complex data samples (I and Q components),
  - no I, Q sample correlation,
  - low inter sample correlation,
  - slow variation of *rms* level with range,
  - slow variation of *rms* level with azimuth.
- signal data size: 5100 lines $\times$ 1250 complex samples (as required to give a 4 look detected image of 512 $\times$ 512 pixels).

SAR Image Data

For the image data encoding studies, the signal data was processed using a precision SAR processor. The image was processed to 4 looks. The characteristics of the SAR image data are as follows:

- SEASAT, 4-look detected, resolution 25m,
- data precision: 12 bits per pixel,
- data entropy: approx 10 bits,
- data type: unsigned word,
- data range: [0,4095],
- data properties:
  - Rayleigh amplitude distribution
  - very high dynamic range:
    - >50 dB for point targets,
    - >30 dB for distributed areas,
  - statistics dominated by speckle:
    - moderate inter-sample correlation,
    - less spectral roll-off than optical sensors,
    - lower SNR than optical data,
- single band,
- image size: 512 $\times$ 512 pixels.

4 Selection of Algorithms

Two encoding algorithms have been used in each part of the investigation:

1. for encoding in the signal domain, JPL's Block Adaptive Quantization (BAQ) algorithm (as used on the Magellan mission to Venus), and Unstructured (full search) Vector Quantization (UVQ) have been compared,

2. for encoding in the image domain, the JPEG DCT algorithm and UVQ have been compared.

The BAQ [KJ89] algorithm was selected for analysis because of its simplicity and because it has given very good results for SAR images obtained on the Magellan mission. The algorithm has been designed specifically for the zero-mean Gaussian statistics of SAR signal data, and is therefore not suitable for the Rayleigh distribution of SAR image data. In its present form, BAQ provides a fixed compression ratio of 4 (*i.e.* from 8 bpp to 2 bpp).

The VQ [Gra84] algorithm was selected because it is the subject of much of the current research in encoding, and it has been shown to give excellent performance on some data sets. The decision to use unstructured VQ was reached because it was believed to give better performance than structured versions, which generally compromise quality for a reduction in "cost" of execution. In retrospect, it would have been better to use a structured version, such as gain/shape VQ, since the cost of execution of UVQ was found to be so high that its full potential was unrealizable, and compromises had to be made, such as working with smaller vectors.

The UVQ algorithm was applied in both the image and signal data domains.

The JPEG DCT algorithm [Wal90] was selected as the second algorithm to be applied in the image domain, since it is becoming a globally accepted still image compression standard. It is recognized that this algorithm is not optimized for SAR data, for example, compression results can be improved using an adaptive DCT algorithm such as that of Chen and Smith [CS77]. The JPEG DCT algorithm was not applied to SAR signal data since the JPEG quantization tables as they stand are not applicable to complex data. SAR signal data has a spectrum similar to that of white noise, with almost as much energy in the high frequency components as there is in the low frequency components, and it is unlikely that the DCT algorithm would produce sufficient energy compaction to enable a useful level of data compression without appreciable loss of image fidelity.

5 Methodology of Study

The activities of the study are broken down into the following steps:

1. Select a suitable operating point *i.e.* compression ratio, and

2. do a full comparison of the algorithms, using all the defined metrics, at the chosen operating point.

To select the operating point for the image data encoding analysis, the UVQ and JPEG DCT algorithms were run with a variety of compression ratios, and SQNR was graphed against compression ratio. The images obtained at each compression ratio setting were viewed, and an operating compression ratio of ≈ 7 was selected, since in this region there was felt to be a small but acceptable

level of distortion in the data. The operating parameters for UVQ were 2×2 vectors and 2^7 codevectors. For JPEG DCT, the quality parameter was set to 8.

For the signal domain encoding, UVQ was run at a variety of compression ratios, and the resulting SQNR compared with that of BAQ at its fixed compression ratio of 4. Since UVQ algorithm produced a similar SQNR to BAQ at the same compression ratio, the operating compression ratio for the signal data study was set at 4. The operating parameters for UVQ for this compression ratio are: 2×2 vectors and 2^8 codevectors.

Three SAR scenes, which provided a variety of scene features, were used for the analysis:

1. Cambridge Bay (sea ice, ocean, snow and arctic tundra),
2. Niagara Falls (urban, roads, river, agriculture, airfield, orchards), and
3. Flevoland (sea, polder, forests, urban).

6 Signal Data Encoding Results

The results of the encoding experiment on the SAR signal data pertaining to the Niagara image, using the BAQ and UVQ algorithms, are presented in Table 1. The Niagara results have been selected for discussion since they accentuate the differences in the results for the two algorithms. The other scenes produced similar results, but the differences were less marked. The reason for this being that the Niagara scene contained greater feature diversity than the other two scenes.

The results for the analysis in the signal (and coding) domain show that BAQ produces a lower SQNR, and in general alters the signal data statistics more then the UVQ algorithm. BAQ does however produce a lower average phase error than UVQ.

Both algorithms altered the appearance of the signal data histograms. For BAQ, the reconstructed histogram shows 4 main peaks, corresponding to the data being mapped to $-GB_2$, $-GB_1$, $+GB_1$, or $+GB_2$ where G is a function of the *rms* level of a block, and the B_i values are chosen to minimize the *rms* error ($B_1 \approx 0.5$, and $B_2 \approx 1.7$). The concentration of the data in 4 main peaks implies that the *rms* level varies very little from block to block. For UVQ, the histograms, which were smooth Gaussian distributions on input, appeared very spiky on output, but still retained the same overall Gaussian appearance. The spikiness is due to the fact that the use of a limited size codebook restricts the output intensity space, *e.g.* by limiting the combinations of intensities of adjacent pixels.

The results for the analysis in the image domain (*i.e.* after processing both original and reconstructed signal data sets) show that the SQNR is again lower for BAQ than for UVQ. In this domain, however, BAQ has degraded the data statistics less than UVQ.

The original, reconstructed and error images for BAQ and UVQ are shown in Figure 2. The reconstructed images for BAQ and UVQ are judged to be of good and equivalent quality. The error images show the error to be fairly evenly distributed for BAQ. For UVQ, the structure of the original image is fairly evident in the error image. This implies that the algorithm is affecting some features more than others. It is seen that the error image is in essence a negative of the original, implying that bright areas have become darker, and dark areas brighter. For BAQ this occurs to a much lesser extent. The phenomenon implies that the radiometric linearity of the coding has been disturbed. This last deduction is corroborated by the fact that the linearity is measured to be 0.98 for BAQ and 0.94 for UVQ, *i.e.* despite the higher MSE value, BAQ has disturbed the linearity

less than UVQ. Preservation of linearity is important, for example, if the data is to be used for SAR sensor calibration.

In terms of the other image domain metrics, both algorithms performed very well, *i.e.* image histograms, spectra, radiometric resolution, speckle and point target characteristics were well preserved (*i.e.* the error is too small to be statistically significant). BAQ did however perform better for wavelength estimation, preserving both the magnitude and frequency of the dominant peaks in the 1D spectra. UVQ preserved the frequency but reduced the magnitude by $\approx$6%.

To summarise, when evaluated in both the signal and image domains, BAQ gave a higher MSE than UVQ. However, since BAQ gave better results in terms of data statistics, radiometric linearity and wavelength estimation, and equivalent results for the remaining metrics, it is deemed to be better than UVQ, with the given operating parameters, for application to SAR signal data encoding.

It is recognized that gain/shape VQ would result in improved linearity, but for the same compression ratio and vector size, this would be at the expense of codebook size. Thus one would expect the overall MSE to be essentially the same, but with the errors more evenly distributed. This last point illustrates another of the shortcomings of MSE: since it is a global measure, it gives no indication of the spatial distribution of the error, and no indication of the extent to which the error affects the various properties important to SAR image analysis.

7 Image Data Encoding Results

The results of encoding experiments on the SAR image data for the Niagara scene, using the JPEG DCT and UVQ algorithms, are presented in Table 2. Again, the Niagara results have been selected for discussion since they accentuate the differences in the results for the two algorithms.

The results for the analysis in the image (and coding) domain show that the SQNR is lower for UVQ than for JPEG DCT, and, in general, UVQ has distorted the data statistics more than the JPEG DCT.

The original, reconstructed and error images for JPEG DCT and UVQ are shown in Figure 3. The reconstructed images for both JPEG DCT and UVQ are judged to be of good quality, though the JPEG DCT reconstructed image is slightly less well focussed than the original (due to attenuation of the high frequency components), and the UVQ reconstructed image shows some loss of contrast (due to the codebook having a more limited intensity space than the original data). The error images show the error to be fairly evenly distributed for both algorithms, with the UVQ error image having slightly more structure than the JPEG DCT error image.

The histogram characteristics were well preserved by the JPEG DCT algorithm. For the UVQ algorithm, although the overall shape of the histogram was preserved, the histogram was much more "spiky" in appearance. Again this is due to the codebook limiting both the intensity space, and the intensity combinations of adjacent pixels.

For both algorithms, the reconstructed spectra appeared very similar to the original spectra, with all dominant peaks reproduced, and no extra peaks introduced. The error spectra did not show evidence of any periodic error effects having been introduced. The error spectra did show that the error for JPEG DCT was greater in the high frequency portion of the spectrum than in the low frequency portion. This was also true for UVQ, but to a very much lesser extent.

Both algorithms preserved the radiometric resolution, and both reduced the speckle component by

≈5%. The radiometric linearity was 1.0 for JPEG DCT, however UVQ was again shown to disturb the linearity, with the slope of the linearity plot being 0.96.

Neither algorithm produced any mis-registration effects, either globally or locally. For almost all the point targets studied, JPEG DCT was found to produce only very slight differences in peak widths, energy and position. UVQ, however, was found to be more likely to alter these characteristics. This is mainly due to the fact that point targets are essentially "rare events" of dimension not much greater than the vectors themselves. A centroid based codebook is not geared to meet the needs of reproducing the individual characteristics of each point target. Point target preservation is important in SAR image analysis.

In terms of wavelength estimation, JPEG DCT preserved both the magnitude and frequency of the dominant peaks in the 1D range and azimuth spectra. UVQ preserved the frequency, but reduced the magnitude by up to 3%. It is however recognized that the peaks studied occurred in the low frequency portion of the spectrum. For higher frequency peaks, JPEG DCT would be more likely to alter the magnitude, since the evaluation of the spectra showed that JPEG attenuates high frequency components more than low frequency components (an inherent property of the JPEG quantization scheme).

To summarise, the JPEG DCT algorithm performed consistently better than the UVQ algorithm. The use of the additional metrics specified in this paper served to highlight the exact way in which the higher error introduced by the UVQ algorithm manifested itself.

Again, the linearity of VQ could be improved using gain/shape VQ. It may also be possible to reproduce the point target characteristics more accurately using some form of classified VQ, as opposed to designing a purely centroid based codebook.

8 Conclusions

In this paper we have suggested a number of evaluation criteria especially suited to determining the manner and extent to which an encoding algorithm distorts SAR data. Both signal and image domain encoding were studied, with two algorithms having been applied in each domain. The results obtained using the given algorithms were described and analysed using the given metrics.

The measurement of MSE or SQNR is important, as it gives a measure of the severity of the unstructured error brought about by encoding. The MSE should always be included as as evaluation metric. However, in order to gain additional insight into the characteristics of the error, and also into the manner of operation of the encoding algorithm, additional metrics are needed.

The metrics which produced the most insight into the manner in which the algorithms acted on the data and the resulting degradations were felt to be:

1. the original versus reconstructed data histograms,
2. error images,
3. error spectra,
4. radiometric linearity, and
5. point target analysis.

The data histograms were useful in both the signal and image domain analysis in showing how the algorithms acted on the data. *e.g.* for BAQ they showed how little the block rms value varied

for the chosen block size. This insight would be useful when selecting a block size. For UVQ the histograms showed the effects of mapping data onto a smaller intensity space and limiting the allowed intensity combinations of adjacent pixels.

The error images, in conjunction with other metrics, such as the data histograms, showed which properties of the original images were giving rise to the highest errors *e.g.* for UVQ it was found that regions with an intensity having a low probability of occurrence, in this case those on the edges of the input intensity space - very bright areas or very dark areas - produced the largest errors.

The spectra highlighted which frequency components were most affected by the coding.

The measurement of radiometric linearity consolidated the results seen in the error image, and showed the severity of this error.

The point target analysis showed how well rare events are reproduced by the given algorithm.

9 Acknowledgments

This work is being performed under ESA Contract No. 9122/90/NL/PR(SC). The authors would like to thank the technical authority, Jean-Luc Marchand of ESTEC, for his advice and direction during the course of this study.

References

[CS77] W.H. Chen and H. Smith. Adaptive Coding of Monochrome and Color Images. *IEEE Trans. on Communications*, COM-25(11):1285–1292, Nov 1977.

[Gra84] Robert M. Gray. Vector Quantization. *IEEE ASSP Magazine*, 1:4–29, Apr 1984.

[Jor80] Rolando L. Jordan. The SEASAT-A Synthetic Aperture Radar System. *IEEE Journal of Oceanic Engineering*, OE 5(2):154–164, April 1980.

[KJ89] R. Kwok and W.T.K. Johnson. Block Adaptive Quantisation of Magellan SAR Data. *IEEE Trans. on Geoscience and Remote Sensing*, 27(4):375–383, July 1989.

[Wal90] Gregory K. Wallace. Overview of the JPEG Still Image Compression Standard. *SPIE Image Processing Algorithms and Techniques*, 1244:220–233, 1990.

Table 1: Evaluation of Signal Encoding Performance (Niagara Scene)

Signal Domain Results	**BAQ**	**UVQ**
CR at operating point	4	4
MSE	107	76
ASQNR (dB)	9.5	11.0
PSQNR (dB)	21.8	23.3
Data range	-20 %	-27 %
Δmean	+24 %	+2 %
$\Delta\sigma$	+8 %	-4 %
Δkurtosis	-8 %	-8 %
Δentropy	-32 %	+7 %
Phase Error (degrees)	18.1	23.2
histograms	4 peaks	spiky
Image Domain Results	**BAQ**	**UVQ**
MSE	9992	7332
ASQNR (dB)	18.6	20.0
PSQNR (dB)	32.2	33.6
Δmean	+8 %	-3 %
$\Delta\sigma$	-1 %	-7 %
Δkurtosis	+0.6 %	+18 %
Δentropy	-0.1 %	-1 %
histograms	preserved	preserved
radiometric resolution	preserved	preserved
speckle	preserved	preserved
radiometric linearity	0.98	0.94
mis-registration	none	none
point targets	preserved	preserved
wavelength estimation	preserved	degraded

Table 2: Evaluation of Image Encoding Performance (Niagara Scene)

Image Domain	JPEG DCT	UVQ
CR at operating point	5.7	6.9
MSE	6208	13430
ASQNR (dB)	21.5	18.1
PSQNR (dB)	34.3	31.0
Δrange	+0.8 %	-6 %
Δmean	-0.04 %	-0.4%
$\Delta\sigma$	-0.1 %	-3%
Δkurtosis	-2 %	-5%
Δentropy	+0.1 %	-19%
histograms	preserved	spiky
radiometric resolution	preserved	preserved
speckle	-5%	-5%
radiometric linearity	1.00	0.96
mis-registration	none	none
point targets	preserved	degraded
wavelength estimation	preserved	degraded

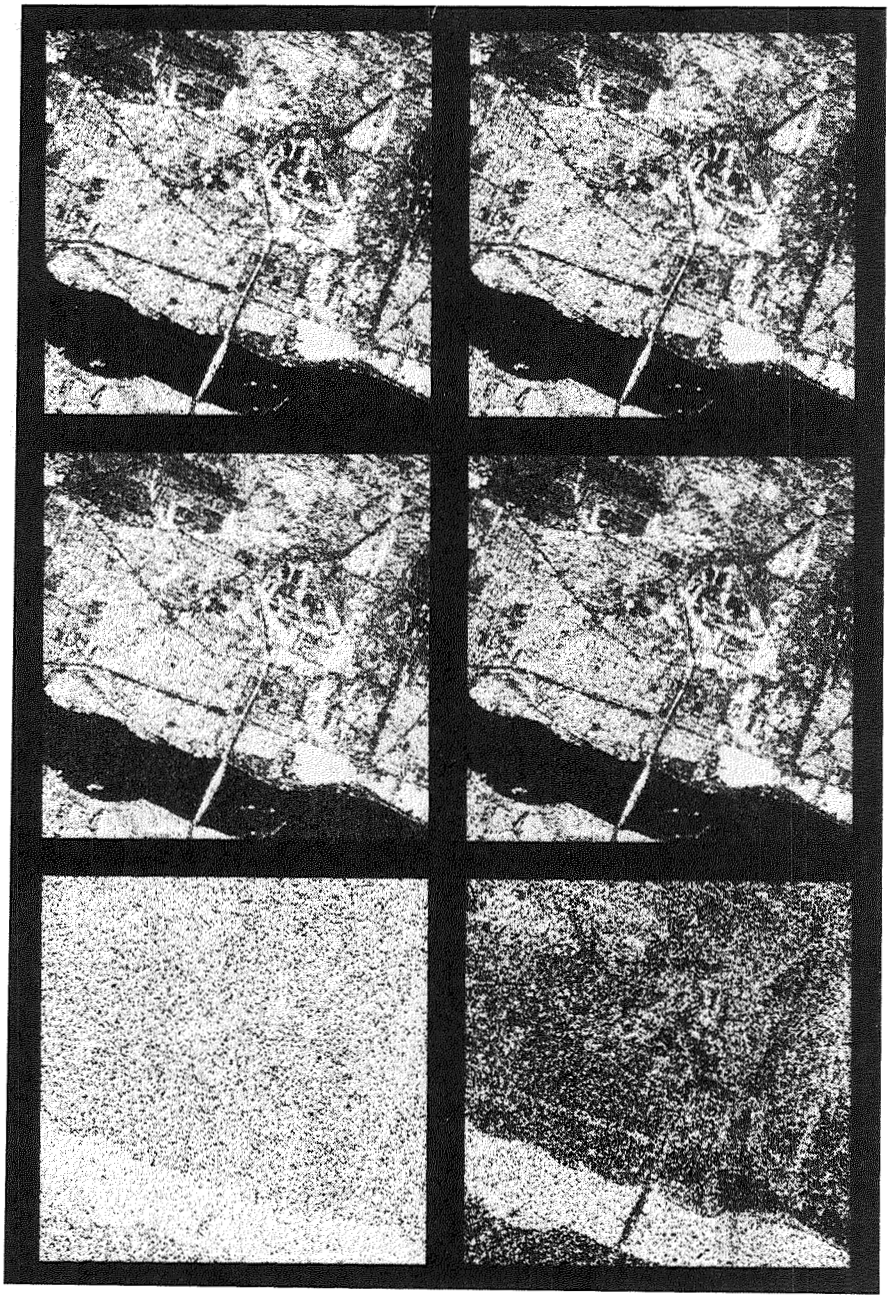


Figure 2: Niagara: Original, Reconstructed and Error Images (Signal Data Encoding)
Top Left: Original; Top Right: Original
Centre Left: BAQ Reconstructed; Centre Right: VQ Reconstructed
Bottom Left: BAQ Error; Bottom Right: VQ Error

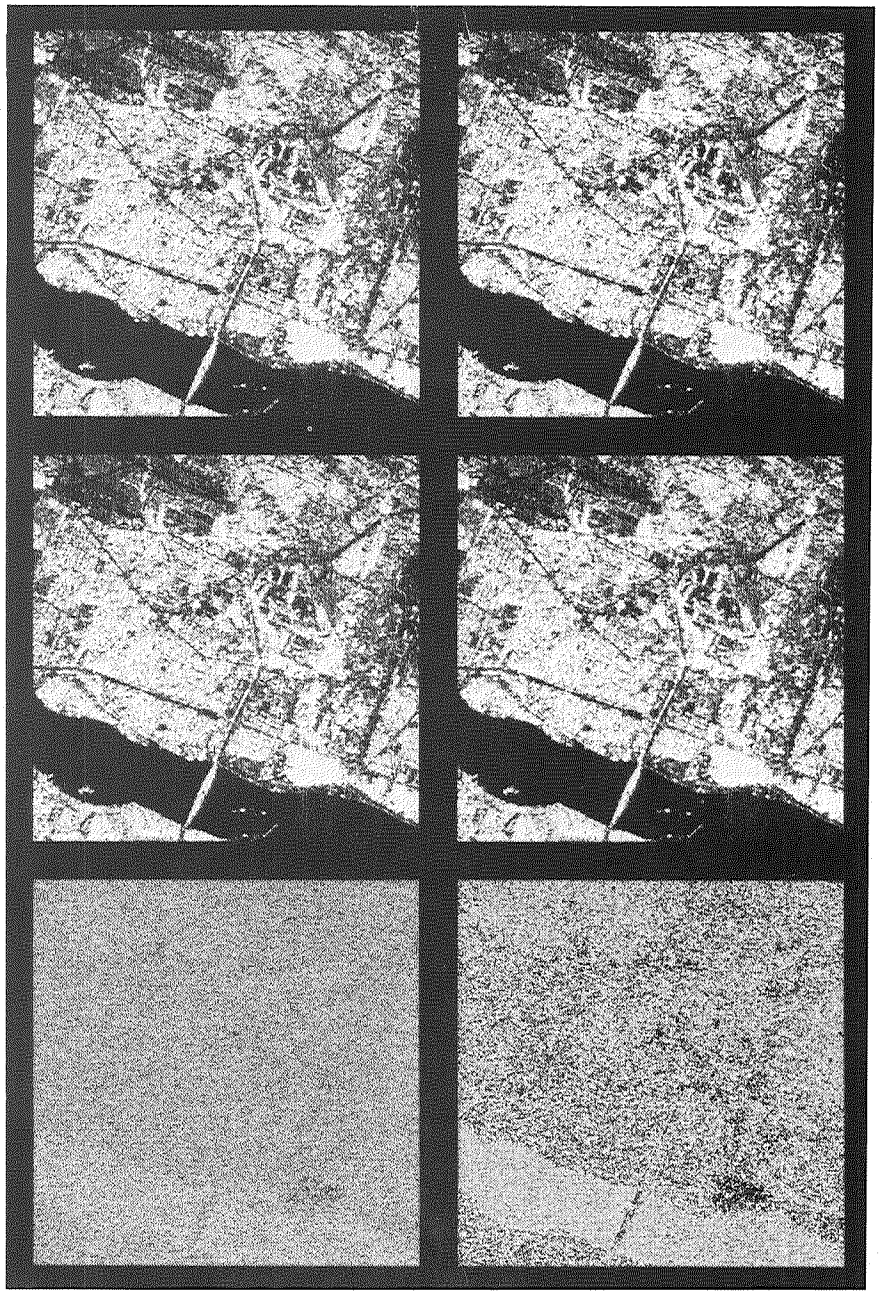


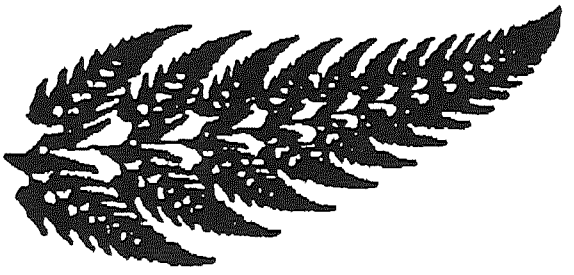
Figure 3: Niagara: Original, Reconstructed and Error Images (Image Data Encoding)
Top Left: Original; Top Right: Original
Centre Left: JPEG DCT Reconstructed; Centre Right: VQ Reconstructed
Bottom Left: JPEG DCT Error; Bottom Right: VQ Error

FRACTAL IMAGE COMPRESSION: A RESOLUTION INDEPENDENT REPRESENTATION FOR IMAGERY

Alan D. Sloan
5550 Peachtree Parkway
Iterated Systems, Inc.
Norcross, Georgia 30092

1. Background

A deterministic fractal is an image which has low information content and no inherent scale. Because of their low information content, deterministic fractals can be described with small data sets. They can be displayed at high resolution since they are not bound by an inherent scale. A remarkable consequence follows. Fractal images can be encoded at very high compression ratios. This fern, for example, is encoded in less than 50 bytes and yet can be displayed at resolutions with increasing levels of detail appearing.



The Fractal Transform was discovered in 1988 by Michael F. Barnsley. It is the basis for a new image compression scheme which was initially developed by myself and Michael Barnsley at Iterated Systems. The Fractal Transform effectively solves the problem of finding a fractal which approximates a digital "real world image."

2. The Fractal Hypothesis for Real World Images

Fractal Image Compression, when used in a lossy mode, provides approximations to a large class of images, termed Real World Images.

- Digital real world photographs contain exceptionally high frequency of occurrence of local large scale correlations, under affine transformations.

- Given a digital real world photograph, it can be approximately partitioned into contractive affine transformations of geometrical structures in the image. Affine transformations of these structures, interpreted as probabilistic events, are unlikely to occur in an arbitrary digital image. However, in a specific image, such occurrences are likely events. For digital real world images, the higher the resolution, the better the approximation.

Affine transformations are composed of translation, differential scaling and differential rotation operations. An affine transform, T, is contractive relative to a given metric, d, when

$$d(T(p),T(p')) < c*d(p,p'), \text{ for some constant } c<1 \text{ and all points p and p'.}$$

Real world images are distinguished from general digital bit-mapped images by the property that they contain local large scale affine correlations unexpectedly often. Foe example, the probability of a randomly generated 1K x 1K x 24 bit/pixel image containing a face of photographic quality is nearly infinitesimal. Experience shows that whenever a real world image contains one face, it tends to contain additional faces. The additional faces are not exact copies of each other, but to a reasonable approximation, can be mapped into each other by affine transformations.

3. Methodology of the Fractal Transform

The Fractal Transform process consists of distinct encoding methodologies. The encoding step begins by creating two collections of images subsets. The first subset consists of Domain Blocks. The collection of Domain Blocks must satisfy two conditions: (a) Domain Blocks must not overlap; and (b) as a whole, the Domain Blocks must exactly cover the image. The second subset consists of Range Blocks. Range Blocks may or may not overlap and may or may not cover the image. Once the subsets of Domain and Range Blocks have been established, a Range Block and a contractive affine transformation is chosen for each Domain Block. The choice is made so that difference between the affine mapping of the Range Block and the Domain Block is minimal. The information which is retained as the code produced by the Fractal Transform consists of the description of the Domain and Range Block sets and, for each Domain Block, the affine transformation and location of the Range Block. No image data need be saved. For each domain block D, only the Range Block address, R_D, and an affine transformation, T_D, are saved.

Decoding an image from its Fractal Transform code consists of repeatedly transforming images starting from an arbitrary initial image, I_O, to create successive images I_1, I_2, ... I_N. I_j is transformed into I_{j+1} by partitioning I_{j+1} into the Domain Block structure obtained during the encoding step and computing the contents of the Range Block region R_D computed from I_j. These computations continue until the difference between I_j and I_{j+1} differ by less than a specified amount.

No inherent scale is introduced during the Fractal Transform process. Relational information and not resolution specific data is retained. Moreover, code produced is of finite length and so of low information content. We conclude that *the Fractal Transform produces a fractal description of images*.

Summary of the Fractal Transform Methodology

- Search for Affine Transformations
    - Partition Image and Create Collection of Image Fragments
    - Map Image into Itself
    - Contractive Mappings
    - Minimize Distance between Image Fragment and Partition Elements
- Produce Code
    - Save the Transformations
    - Discard All Image and Other External Data
    - Organize Transformations into Fractal Image Format
- Reconstruct Image Algorithmically
    - Iterative Process
    - Repeated Application of Image Processing Transformation

FRACTAL TRANSFORM - TECHNICAL

Step 1 - Partition the image to be encoded creating Domain Blocks.

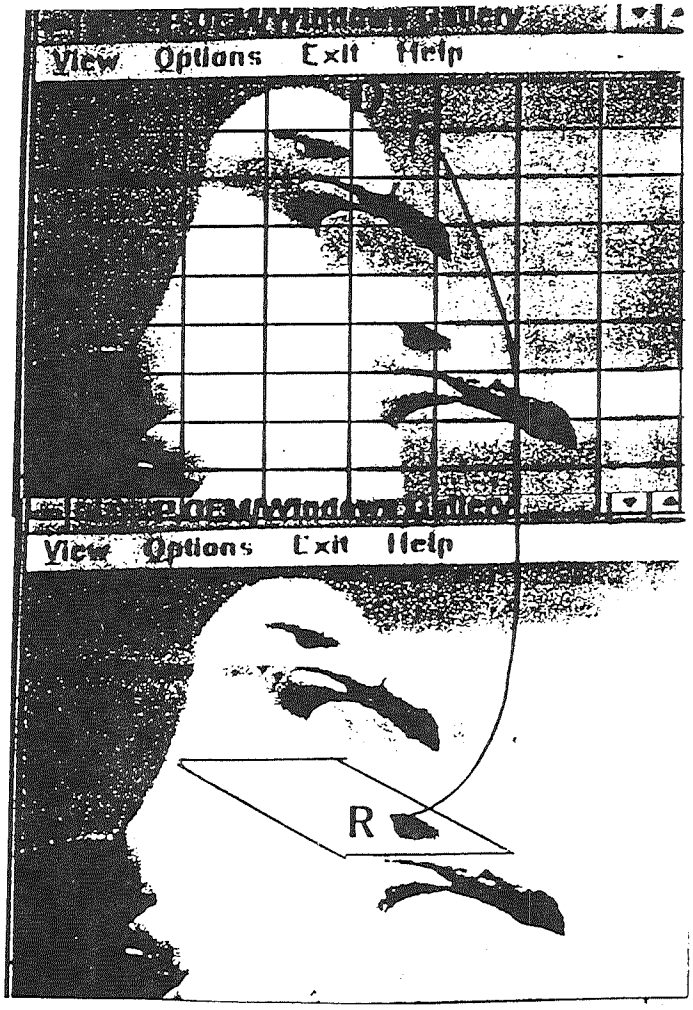
Step 2 - Choose another collection of image subsets, called Range Blocks.

Step 3 - For each Domain Block, D, choose Range Block R_D

and contractive affine transformation $T_D : R_D \rightarrow D$

so that image distance between D and $T_D(R_D)$

is minimal.



FRACTAL TRANSFORM CODE CONSISTS OF

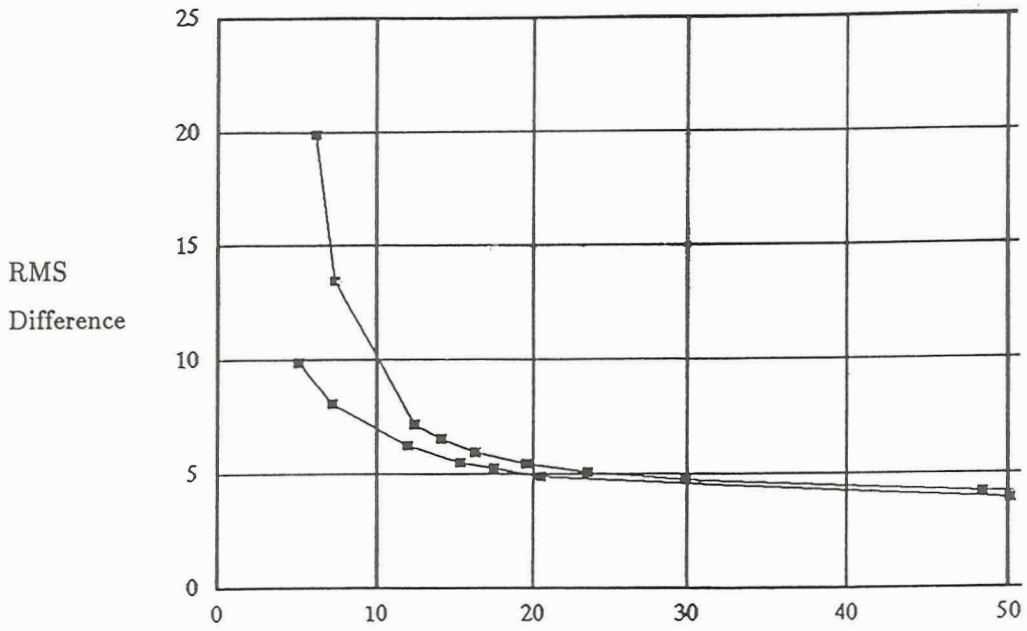
Domain Block Set,
Range Block sets, and
for each Domain
Block,
the affine
transformation
and location of
the Range Block.

4. Comparison with the Discrete Cosine Transform

Study I

Twenty images input at a resolution of 640 x 400 x 24 bits/pixel were compression by the Fractal Transform and a JPEG implementation of DCT compression. The image set includes data from 9 ISO standard images. Image fidelity is measured by comparing the root-mean-square (RMS) difference between the original digital image and the compressed and then decompressed image. Compressed file sizes range from about 5K to 50K. Typically, for larger file sizes the image fidelity of the techniques are comparable. However, at lower file sizes, the Fractal Transform yields significantly better quality imagery than the DCT technique, as measured by the RMS.

The following data is from the "balloons" ISO image:



"Balloons" ISO Image.

Vertical Axis is RMS Difference between Compressed and Original Data.

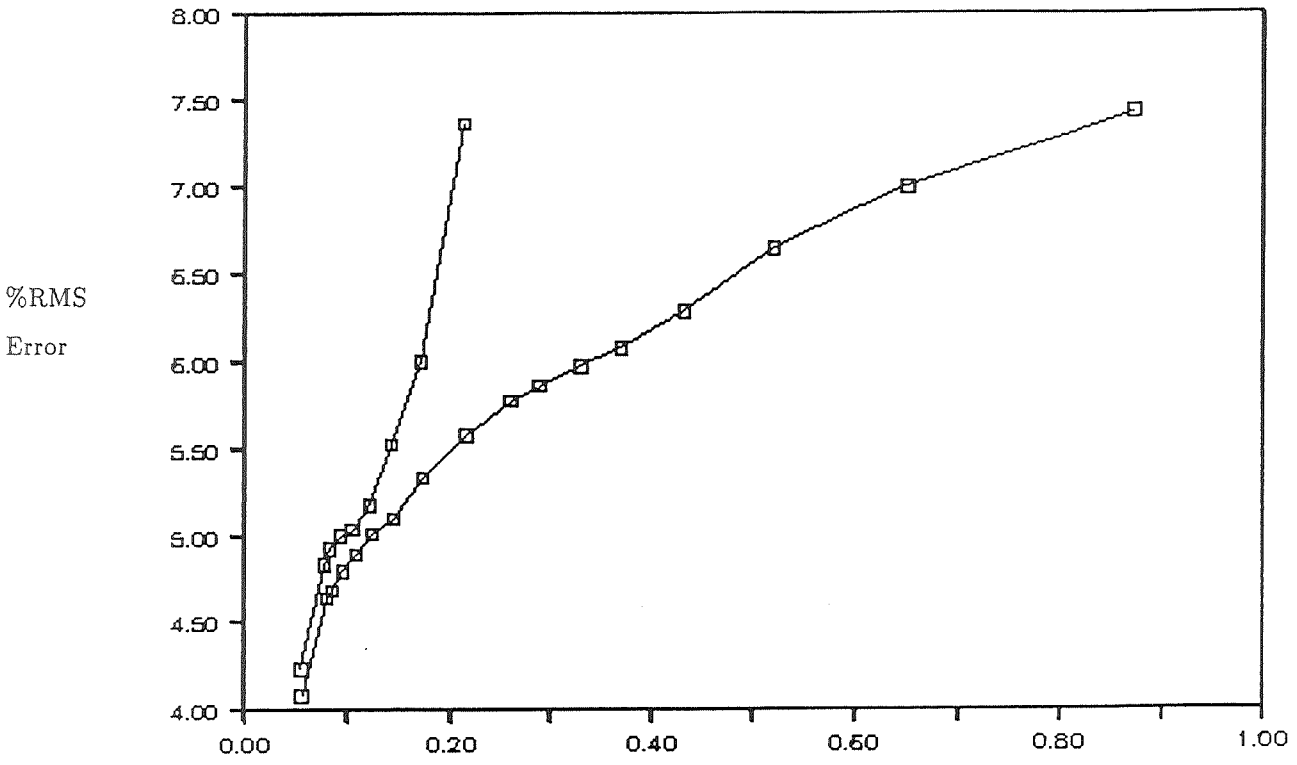
Lower Curve is Fractal Transform Data. Upper Curve is DCT Data.

Vertical Axis is RMS difference. Horizontal Axis is Compressed File Size in Kilobytes.

Study II

Two 5 band AVHRR images were obtained at resolution 1024 x 1024 x 16 bits/pixel for each band. The data pre-processed to retain 8 bits of data. This pre-processed data was compressed by Fractal Compression technology and by a JPEG implementation of DCT by Xing. The data below represents compression of one of the two five band images resulting in file sizes ranging from about 90,000 bytes to 6,000 bytes.

The vertical axis provides values for % root-mean-square error(%RMS). The RMS is computed for the five band image and is then divided by 255, since the RMS can vary between 0 and 255. Using the RMS as a measure of fidelity, higher %RMS values indicate less fidelity. The horizontal axis measures the (compression ratio)/1000, which varies inversely as the file size of the compressed image.



The upper and shorter curve describes the JPEG results. The lower and longer curve describes the results of Fractal Compression on the same digital imagery. The rapid increase in %RMS for the JPEG technique points towards the breakdown of the JPEG methodology. This is borne out by visual inspection in which the JPEG images reveal a blocky nature, the characteristic artifact which results as the limit of the JPEG technique is approached. The Fractal Compression technique permits much smaller file sizes to be reached.

5. Resolution Independence

A straight line in encoded in a computer by a formula, not by a bit map image of a representative line. Output and display questions are, for the most part, independent of the formula for the line. When an output device is attached to a computer, a software program which targets the output specific format (resolution, aspect ratio, pixel/depth, number of colors, etc.) of the output device, is used to generate a suitable display from the resolution independent formula which represents the line. Analogously, the Fractal Image Format provides a description of an image which is independent of the output device.

The Fractal Image Format is a Resolution Independent Representation for Images.

No inherent scale is introduced during the Fractal Transform process. Relational information and not resolution specific data is retained. Therefore Fractal Transform compression produces a resolution independent description of images.

An Evaluation of Resolution Independence

Forty 640 x 400 x 24 bits/pixel satellite images of the earth were compressed to a variety of file sizes between 15K and 5K by both DCT and Fractal Transform technologies. In the case of DCT the original image was compressed and decompressed and a RMS error was computed between the decompressed and original image. To evaluate the nature of the resolution independence of the Fractal Transform technique, each of the 640 x 400 images was first subsampled to create a 320 x 200 image. This smaller image was compressed and then decompressed at the larger 640 x 400 resolution. A RMS error was computed for the 640 x 400 original and fractal zoom image. In each case, the Fractal Zoom RMS was no worse than the DCT RMS. On average, the Fractal Zoom technique reduced RMS errors by 15% from those produced by DCT.

THE IMPLEMENTATION OF A LOSSLESS DATA COMPRESSION MODULE IN AN ADVANCED ORBITING SYSTEM: ANALYSIS AND DEVELOPMENT

Pen-Shu Yeh†, Warner H. Miller†, Jack Venbrux‡, Norley Liu‡, Robert F. Rice*
†Goddard Space Flight Center, Greenbelt, MD 20771
‡Microelectronics Research Center[1], University of Idaho, Moscow, ID 83843
*Jet Propulsion Laboratory, Pasadena, CA 91109

Abstract

Data compression has been proposed for several flight missions as a means of either reducing onboard mass data storage, increasing science data return through a bandwidth constrained channel, reducing TDRSS access time, or easing ground archival mass storage requirement. Several issues arise with the implementation of this technology. These include the requirement of a clean channel, onboard smoothing buffer, onboard processing hardware and on the algorithm itself, the adaptability to scene changes and maybe even versatility to the various mission types.

This paper gives an overview of an ongoing effort being performed at Goddard Space Flight Center for implementing a lossless data compression scheme for space flight. We will provide analysis results on several data systems issues, the performance of the selected lossless compression scheme, the status of the hardware processor and current development plan.

1 Introduction

Before implementing a data compression scheme onboard a spacecraft, it is important to address issues in the telecommunication channel and the architecture of the data system in which it resides. The advanced orbiting systems of the 1990's and beyond will demand a communication network which can support a wide range of data rates, complex international constellations of space platforms, extensive onboard computer networking and possibly cross-support among missions. To meet the requirement of such a network and data system, the Consultative Committee for Space Data Systems (CCSDS) has published a recommended standard: "Advanced Orbiting Systems, Networks and Data Links: Architectural Specification" [1], to provide descriptions of the architecture of a network and data structure recommended for future orbiting platforms.

An important feature of such a data system is that all sensor data are packetized into the hierarchical structure shown in Fig. 1. Sensor data are first encapsulated into a CCSDS packet of length up to 2^{16} bytes. It is further multiplexed with other CCSDS packets originated from other paths into Multiplexing Protocol Data Unit (MPDU) of fixed length. After being padded with error protection bits and other inserted data, the MPDU is further assigned a Virtual Channel, and converted into a Virtual Channel Data Unit (VCDU). Again, this is a fixed length data unit.

[1]Supported in part by NASA under grant NAGW-1406

The subsequent stage which takes MPDUs from both single and multiple sources is the Virtual Channel Access (VCA). Its output data rate is the fixed link data rate assigned to the platform. The VCA may accept inputs from various data sources of variable data rates and bandwidths. To provide a constant output data rate from VCA, a smoothing data buffer is required and occasionally fill VCDUs are transmitted when the buffer experiences data underflow.

An efficient lossless data compression scheme will almost always produce variable length coded bits. These coded bits will be concatenated to form CCSDS packets, which consequently will be of variable lengths too. This requires that the multiplexing unit be provided with smoothing buffer in order that the output MPDU rate is constant. Associated with the selected buffer size, there will also be instances when a fill MPDU is necessary. A filled data unit can be regarded as a decrease in the channel utilization or efficiency.

In the sequel, we will first address data system issues related to implementing onboard data compression. A description and performance analysis of a selected lossless compression algorithm will be given. It will be followed by the results of an ASIC development of this algorithm. Finally, a brief summary of our current efforts will be given.

2 Systems Issues

2.1 Clean Channel Requirement

As pointed out in [2] of the last Data Compression Workshop held at Snowbird, April 11, 1991, one characteristic of compressed data is its sensitivity to noise. That is, one bit in error can result in a burst of data errors during the reconstruction process. This sensitivity to noise results from the fact that most compression algorithms reconstruct data based on values from more than one sample. Specifically, it is apparent that for algorithms which perform Differential Pulse Code Modulation (DPCM) as a front end process, a reconstruction error will tend to propagate to the end of a packet. In general, a tradeoff exists between choosing a suitable packet length to match the telecommunication channel characteristics and the ease of interfacing instruments within a data system. However, the channel coding recommended by CCSDS employing a concatenated error control coding scheme of an outer Reed-Solomon (255, 223) code and a rate 1/2 convolutional inner code [3], will provide a channel with bit error rate (BER) much lower than 10^{-9} at SNR of 3 db. Operational use of this concatenated system should typically yield even lower error rate, far lower than the stated requirement of 10^{-6} for the compressed data [2].

2.2 Buffer Location, Requirement and Channel Efficiency

Lossless data compression methods, by which redundancy is removed from the source data, result in variable length bit strings which can be packetized. The variable length CCSDS packets are first enclosed in fixed length MPDUs. These MPDUs are input to the VCA either synchronously or asynchronously as shown in Fig. 2. For the synchronous sampling by the VCA: a MPDU packet consisting of either valid CCSDS packets or fill bits is passed to the VCA, at every sampling period t_s. In this scheme, the smoothing buffer is provided at the MPDU generator location. For the asynchronous sampling scheme: a MPDU is

provided to the VCA only when it is filled with valid CCSDS packets. Therefore, the input to the VCA is sampled at variable time intervals which are a multiple t_p, the CCSDS packet generation period. The constant downlink data rate is achieved by providing a buffer at the VCA. The system sampling time t_s is determined by the data rate allocated to a specific instrument, while the packet generation period t_p relates to a sensor's data collection scheme.

During data underflow, filled MPDUs for Fig. 2(a), or filled CVCDUs for Fig. 2(b), are generated to maintain constant link rate. This causes reduced channel utilization.

For the smoothing buffer, its size requirement depends on the packet statistics. These effects have been simulated in a study [4] which shows that the long-term performance of both sampling strategies in Fig. 2 are similar. That is, the maximum buffer requirement and the channel efficiency are comparable. An example is given in Fig. 3, which shows these effects as a function of the sampling ratio t_s/t_p. The result was obtained by assuming a packet source of Gaussian distribution of mean packet length 1 MPDU and a σ of 0.1, which is related to the variation in source statistics. The performance is characterized in terms of the buffer length requirement (in MPDUs) and average fill fraction.

2.3 Recoverability

As mentioned earlier, a channel error on the compressed data bits is likely to cause reconstruction error that will also affect subsequent reconstructed data. This type of error propagation can be limited to the error within a single packet by employing a data compression scheme that resets at the beginning of every packet.

3 An Adaptive Lossless Source Coding Algorithm

In selecting a lossless compression algorithm for onboard applications, several criteria are considered:

Adaptivity: Spacecraft sensor data are usually characterized by wide variation in the statistics. Representative data come from Earth observation data over clouds, ocean, land, or spectrograph data of solar activities, or star fields, or galaxies. A selected algorithm should compress data at near optimal rate when the scene changes (even for one instrument) to fully exploit the benefit of data compression.

Ease of Implementation: For onboard implementation, an algorithm should require few processing steps, small memory, and insignificant power.

The universal source coding scheme, devised by Rice [5] [6] [7] [8] [9] was selected. Its function and performance are described in the following.

3.1 The Universal Source Coding Scheme

The Rice algorithm is a structure that provides efficient performance over a broad range of source entropy. This universality is accomplished by adaptively selecting the best of several

options of an easily implemented variable length coding algorithm on the basis of a block of input samples. The size of the block is a compromise between algorithm adaptivity and the necessary overhead bits to identity algorithm options. Our earlier study has shown that a block size of 16 samples is optimal for most our test imagery. This block of input samples is pre-processed by first performing DPCM (or higher order prediction) and a mapping of the data into non-negative integers. A block diagram of the algorithm structure is provided in Fig. 4. One option of the algorithm codes these integers with a comma code, the other options are obtained by splitting a specified number of the least significant bits, k, off the integers, to be appended later to the comma code of the remaining most significant bits. These options are considered as coders running in parallel. The one that produces the least number of coding bits is selected and ID bits are generated to signal this option to the decoder.

3.2 Optimality of The Compression Algorithm

In an earlier analysis [10], it was shown that for source symbol sets having a Laplacian distribution, the first option is equivalent to a class of Huffman code under the Humblet condition. The other split-sample options are shown to be equivalent to the Huffman codes of a slightly modified Laplacian symbol set, at integer symbol entropy levels. For NASA's applications, especially on imagery, for which the symbol probabilities after DPCM are well modelled as Laplacian, the practical result is simple and profound: the Huffman code to use at each integer entropy value $(k+2)$ is the corresponding k split-sample option.

The theoretical performance of these split-sample options on a Laplacian symbol set is given in Fig. 5. As more split-sample options are included in the coding structure, the performance curve will be extended in the upper-right direction following the same trend.

A major advantage of this coding structure is that the codeword for each symbol is completely specified by knowing its order in the integer symbol set. No codebooks are needed, this significantly simplifies onboard hardware implementation.

3.3 Simulation and Comparison with Other Techniques

A set of nine test imagery of 128x128 pixels, acquired from NASA image library and shown in Fig. 7, was compressed using the selected algorithm. The top two rows are 8-bit data while the bottom row has 12-bit AVIRIS test data. The results are given in Fig. 6. The efficacy of the algorithm is clearly demonstrated.

In order to compare with other techniques, four other samples of University of Southern California (USC) 8-bit test images, shown in Fig. 8 are used. The results are listed in Tables 1, 2 and 3 in terms of three performance parameters: percentage reduction, the compression ratio (CR), and total coding bytes. For the Ziv-Lampel (LZ) algorithm, the *compress* utility on UNIX system is used [11]. The *pack* utility simulates the adaptive Huffman (AH in the Tables) code. The arithmetic coding (denoted as ARi in the Tables) scheme is implemented using [12]. To provide a fair basis for comparison, we also include results obtained by using these techniques on the same pre-processed, *i.e.* DPCM+mapping, imagery. It is expected that this pre-processing will largely de-correlate data and increase the performance of the three other techniques with which we are comparing the Rice algorithm. For the LZ, AH

and ARi techniques, the results are listed under columns marked as p+LZ, p+AH and p+ARi.

It should be noted that the results for the Rice algorithm include an 8-bit reference for every scanline and a 3-bit ID for every 16 samples.

4 ASIC Development

An Application Specific Integrated Circuit (ASIC) chip set has been designed, fabricated and tested to perform the selected universal lossless compression algorithm [13] [14]. The general architecture follows Fig. 4.

4.1 General Descriptions of the Chip Set

The algorithm lends itself to a high speed integrated circuit implementation because:

1. The encoding process allows a highly pipelined architecture, and most of the decoding process can be pipelined as well.
2. Hardware can be shared inside the chips because the options are similar in structure.
3. No external RAM is needed to store tables or statistics.
4. No lookup tables are required on either the encoder or the decoder. The total on-chip RAM is only 320 bytes.

To allow easy interface with an onboard data system such as depicted in Fig. 2, the coded bits are preceded by a header word containing the total number of coding bits. It will be stripped by the packetizer before being concatenated with other blocks into CCSDS packets.

The default DPCM uses the previous value as a predictor, however, the design also allows an external reference to be used as predictor. The pre-processor functional module can also be by-passed completely to allow using only the entropy coding module. Because the encoder is designed to be able to operate continuously at the sample frequency, no sample buffer is needed to store scanlines. Features are also built in the decoder to prevent any decoding error induced by the channel noise to propagate to the next packet.

In order to adapt to a variety of potential instruments, the current chip set is designed to handle 4-14 bits of digital data. A 4-bit ID is attached to every block of 16 coded samples. In addition, reference samples can be inserted at a user-specified interval.

4.2 Chip Set Parameters

The chip set has been designed in a 1 micron CMOS process for low power consumption and high data rate. The resulting die area for both encoder and decoder was only 5mm on a side. Fig. 9 shows the chip plots for the encoder and decoder.

The designed chip set was fabricated and tested on a Hewlett Packard HP82000 IC tester with parametric tests and functional tests that use over 100,000 vectors. Table 4 lists its parameters.

4.3 Flight Readiness

The chip set, named Universal Source Encoder (USE) and Universal Source Decoder (USD), was fabricated using the Hewlett Packard commercial process line which was tested to withstand a total radiation dose of up to 1 Mrad. The USE chip will undergo thermal cycles and a vibration test as part of chip qualification before possible launch. Meanwhile, a rad-hard version of the USE/USD chip set will be developed before being installed in the flight data system.

5 Current Development Plan

Currently, a testbed for the USE chip is being designed. This testbed includes packetizer, multiplexer and interface to VCA on the encoder side. Plans have been made to perform end-to-end test through NASA's TDRSS and the NASCOM system, where the USD chip will be located to decode compressed data.

Acknowledgement

The authors would like to acknowledge the following personnel at Goddard Space Flight Center for participating in the design of the testbed: Wai Fong, Joe Miko, Karen Michael, Caleb Principe. Thanks are also due to Raghu Srinivasa for implementing the arithmetic coder.

References

[1] *Advanced Orbiting Systems, Networks and Data Links: Architectural Specification* CCSDS 701.0-B-1, Oct. 1989. Availabe from: CCSDS Secretariat, Communications and Data Systems Division (code-TS), National Aeronautics and Space Administration, Washington, DC 20546.

[2] W. Tai, "Data compression - the end-to-end information systems perspective for NASA space science missions," *Proc. of the Space and Earth Science Data Compression Workshop*, Snowbird, Utah, 1991.

[3] *Advanced Orbiting Systems, Networks and Data Links: summary of concept, rationale and performance* CCSDS 700.0-G-2, Oct. 1989.

[4] P. S. Yeh and W. H. Miller, "The implementation of a lossless data compression module in an advanced orbiting system: issues and tradeoffs," presented at First ESA Workshop on Computer Vision and Onboard Processing, The Netherlands, 1991.

[5] R. F. Rice and J. R. Plaunt, "Adaptive variable-length coding for efficient compression of spacecraft television data," *IEEE Trans. on Communication Technology*, Vol. COM-19, No. 6, pp. 889-897, 1971.

[6] R. F. Rice, "Some practical universal noiseless coding techniques," JPL Publication 79-22, Jet Propulsion Laboratory, Pasadena, California, 1979. Available from: JPL Publication Office, Jet Propulsion Laboratory, Mail Stop 111-130, Pasadena, CAlifornia 91109.

[7] R. F. Rice and J.-J. Lee, "Some practical universal noiseless coding techniques, part II," JPL Publication 83-17, Jet Propulsion Laboratory, Pasadena, California, 1983.

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

[9] R. F. Rice, "Practical universal noiseless coding techniques, part III," JPL Publication 91-3, Jet Propulsion Laboratory, Pasadena, California, 1991.

[10] P. S. Yeh, R. F. Rice and W. H. Miller, " On the optimality of code options for a universal noiseless coder," *JPL Pub. 91-2*, 1991.

[11] T. A. Welch, "A technique for high-performance data compression," *IEEE Computer* Vol. 16, No. 6 June 1984.

[12] I. H. Witten, R. M. Neal and J. G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM,* V. 30, N. 6, June 1987.

[13] J. Venbrux, N. Liu, K. Liu, P. Vincent and R. Merrel, "A very high speed lossless compression/decompression chip set," JPL Publication 91-13, Jet Propulsion Laboratory, Pasadena, California, 1991.

[14] J. Venbrux, P. S. Yeh and N. Liu, "A VLSI chip set for high speed lossless data compression," accepted for publication in *IEEE Trans. on Circuits and Systems for Video Technology*, 1992.

C-2

CCSDS DATA UNIT STRUCTURE

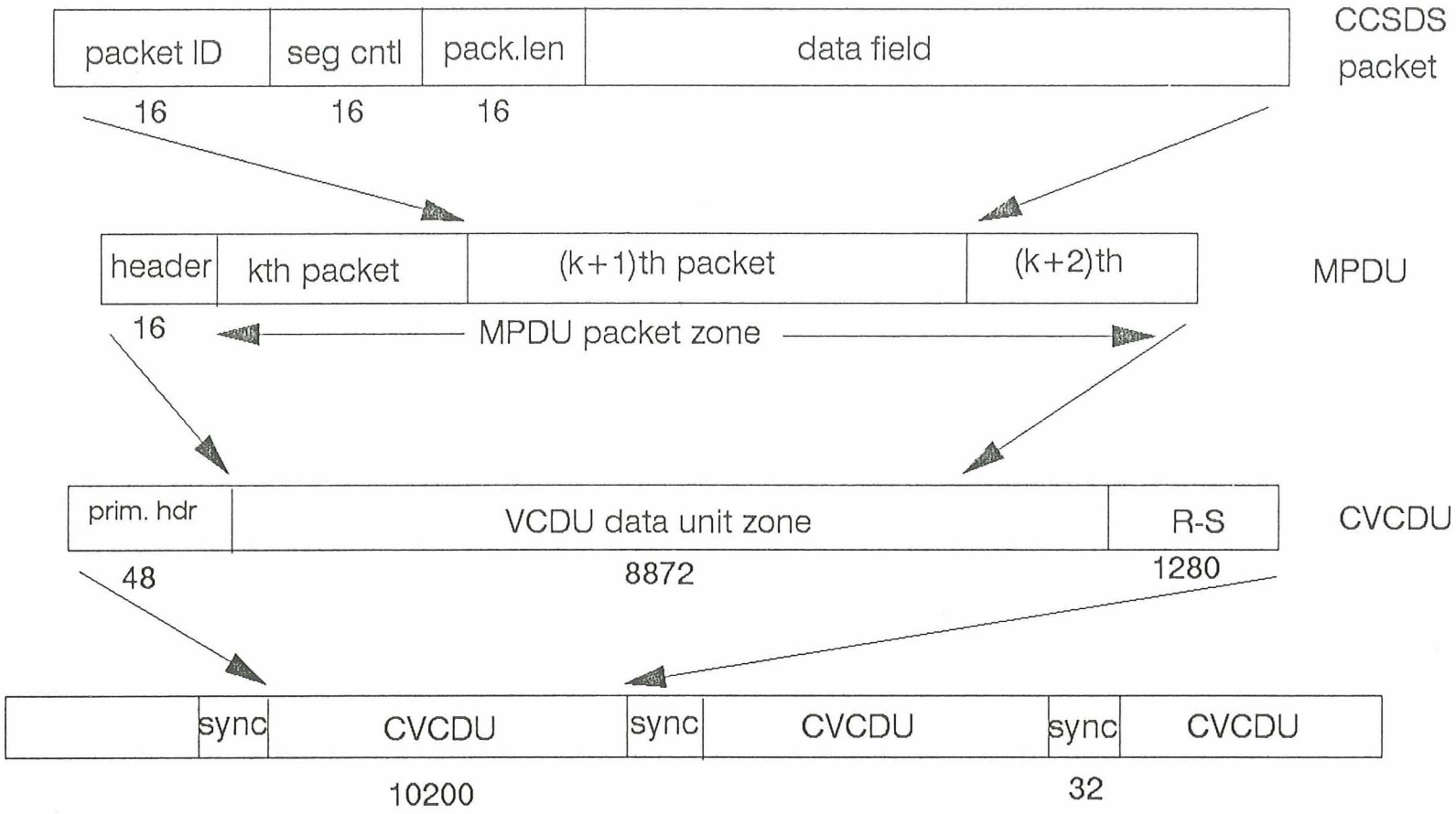


Fig.1 CCSDS packet data structure

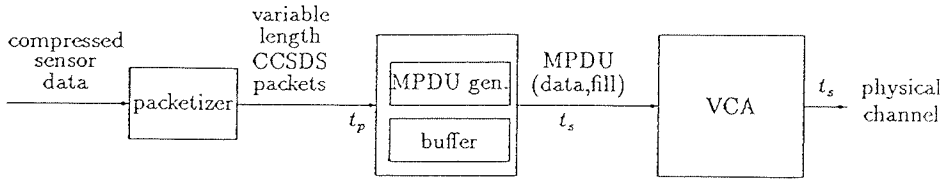


Fig.2(a) Synchronous packet data flow

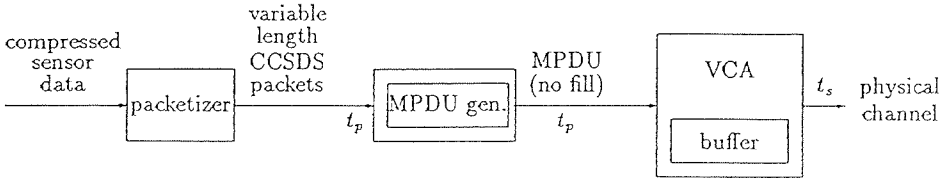


Fig.2(b) Asynchronous packet data flow

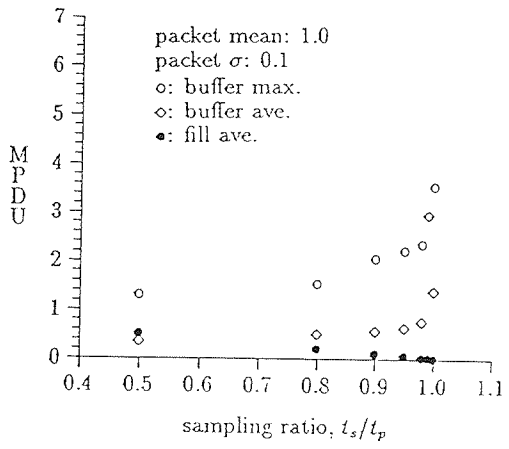


Fig.3(a) Performance of 2(a)

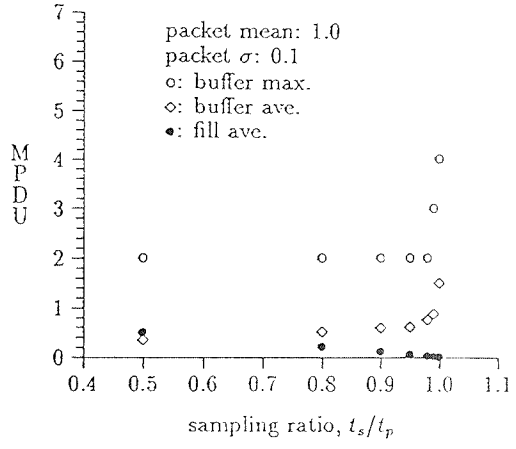


Fig.3(b) Performance of 2(b)

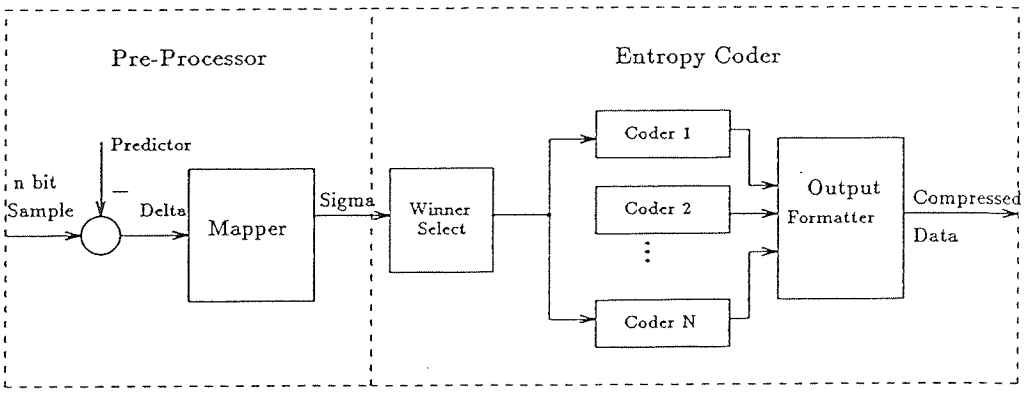


Fig.4 The Rice algorithm architecture

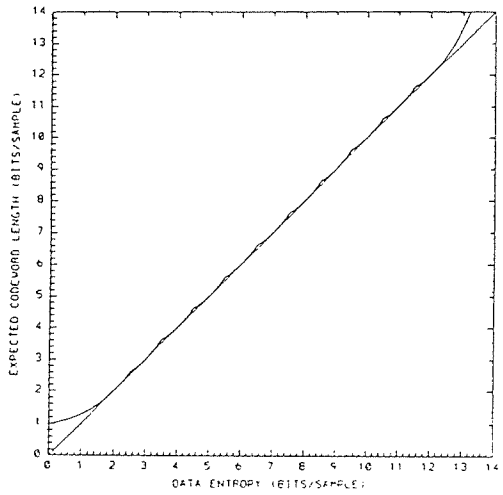


Fig.5 Theoretical performance of the split-sample options

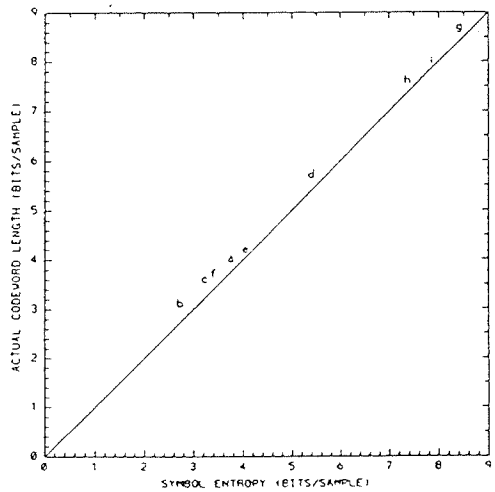


Fig.6 Performance of the coder on samples of 9 aerial imagery

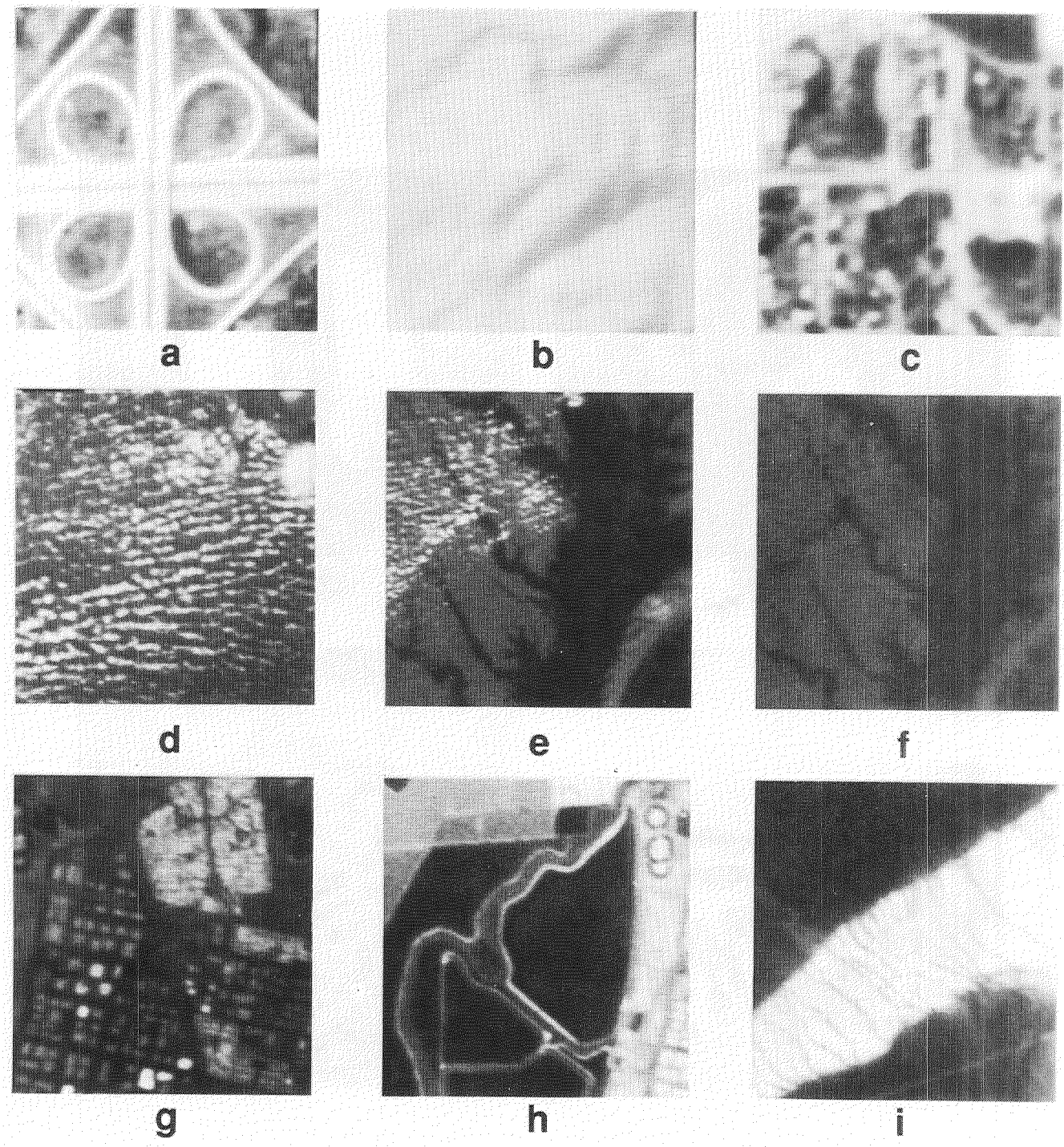
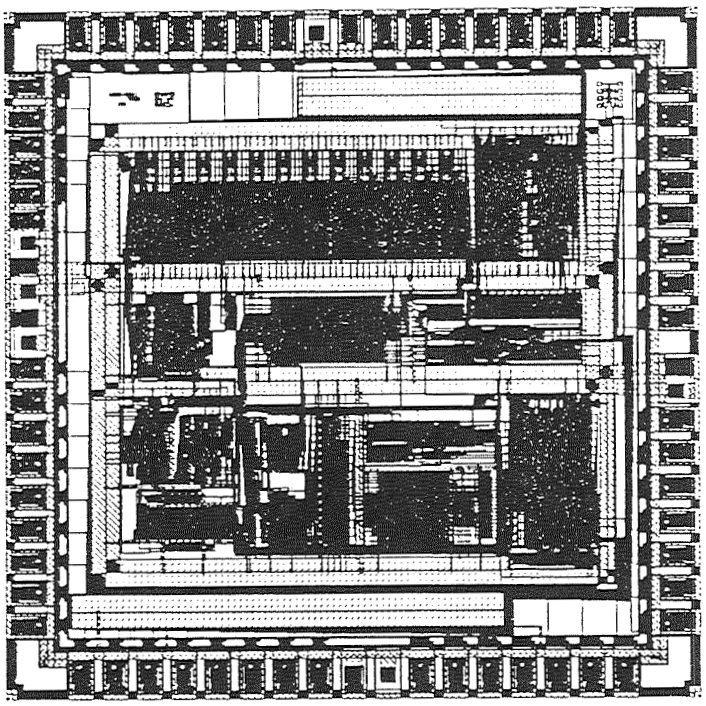
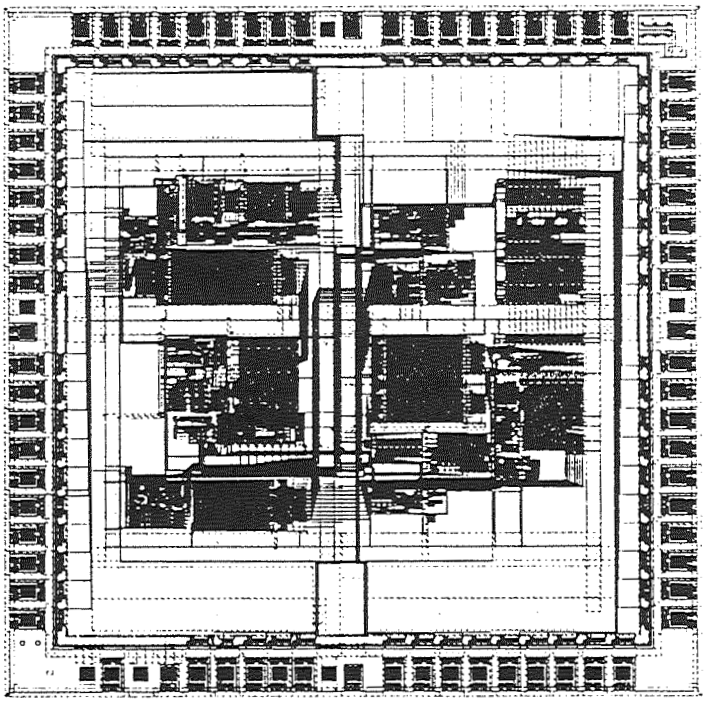


Fig. 7 The 9 GSFC test aerial images

Fig.8 The 4 USC test images



(a) Encoder Layout.



(b) Decoder Layout.

Fig. 9 The chip set layout

	LZ	AH	ARi	p+LZ	p+AH	p+ARi	Rice
girl	28.48	18.80	21.50	30.02	39.40	40.23	41.50
baboon	-4.79	5.90	7.75	-0.15	17.80	18.76	18.44
F16	22.32	14.50	20.76	32.04	40.20	41.65	42.36
aerial	9.42	12.10	13.72	17.63	29.10	29.63	29.95

Table 1: Percentage reduction of files after compression

	LZ	AH	ARi	p+LZ	p+AH	p+ARi	Rice
girl	1.40	1.23	1.27	1.43	1.65	1.67	1.71
baboon	0.95	1.06	1.08	1.00	1.22	1.23	1.23
F16	1.29	1.17	1.26	1.47	1.67	1.71	1.74
aerial	1.10	1.14	1.16	1.21	1.41	1.42	1.43

Table 2: Compression ratio of each test image

	LZ	AH	ARi	p+LZ	p+AH	p+ARi	Rice
girl	46867	53202	51443	45857	39724	39178	38339
baboon	-	246587	241760	-	215575	212976	213816
F16	203621	224052	207697	178132	156865	152945	151091
aerial	237443	230318	226247	215920	185882	184455	183639

Table 3: Total number of coding bytes after compression

	Encoder	**Decoder**
Designed Frequency	20 Mhz	20 Mhz
Design Specs (wc process)	125C,4.5V	70C,4.75V
Measured Lab Freq	50+ Mhz	50 Mhz
Lab Bit Rate N=14	700+ Mbits	350 Mbits
Power(20Mhz,5.5V,100pF+)	0.34 W	0.24 W
Transistors	36,487	33,451
Die Size	5mm X 5mm	5mm X 5mm
Process	1.0um CMOS	1.0uM CMOS
Package	84 pin PLCC	84 pin PLCC

Table 4: Chip Set Summary

LOSSLESS COMPRESSION OF IMAGE DATA PRODUCTS ON THE FIFE CD-ROM SERIES

Jeffrey A. Newcomer
Hughes STX Corporation
4400 Forbes Blvd.
Lanham, Maryland 20706-9008

Donald E. Strebel
VERSAR Incorporated
9200 Rumsey Road
Columbia, Maryland 21045

Abstract. How do you store enough of the key data sets, from a total of 120 gigabytes of data collected for a scientific experiment, on a collection of CD-ROMs, small enough to distribute to a broad scientific community? In such an application where information loss in unacceptable, lossless compression algorithms are the only choice. Although lossy compression algorithms can provide an order of magnitude improvement in compression ratios over lossless algorithms, the information that is lost is often part of the key scientific precision of the data. Therefore, lossless compression algorithms are and will continue to be extremely important in minimizing archiving storage requirements and distribution of large earth and space (ESS) data sets while preserving the essential scientific precision of the data.

Data preservation, distribution, and archiving were integral and essential elements for the satellite, aircraft, and field data collected by the FIFE (First ISLSCP[1] Field Experiment) in 1987 through 1989 over the Konza Prairie area near Manhattan, Kansas. In total, the raw and the derived data products comprise approximately 120 gigabytes of which the image data comprise over 99%. An important element of the planned final archive is a set of CD-ROMs which will contain a reduced data set felt to satisfy the primary objectives of the experiment. In order to store as much key image data as possible on the CD-ROMs and to preserve the scientific precision of the data, a lossless compression algorithm was devised. Use of the algorithm on AVHRR-LAC, Landsat TM, SPOT, NS001, and ASAS image products has resulted in an average compression ratio of 2:1 with ancillary and supporting files of information having ratios as large as 16:1.

The ordered steps of the algorithm include : 1) normalization of the columns of the data matrix, 2) normalization of the rows, and 3) viewing the remaining values in each row as layers of bits that are either runlength encoded or packed back into bytes depending on the calculated average runlength of the line. The compression package, containing functions for processing 8, 16, and 32 bit values and processing control, is written in C and is operational on a VAX computer system in the Laboratory for Terrestrial Physics at NASA GSFC. A complementary set of image restoration software was developed and can be run on a wide range of computing platforms from PCs to workstations.

We envision this intuitive compression algorithm to be useful on a broad range of spatial data sets including gridded layered modeling data sets such as terrain, spectral, and meteorological variables that will be required for coordinated earth systems field experiments during the next decade.

1. Introduction

Distributing large amounts of scientific data such as satellite and aircraft imagery, mandates some form of data compression to minimize storage and data transmission. Although lossy compression algorithms can provide compression ratios of up to 100:1, the precision that is lost

[1]International Satellite Land Surface Climatology Project

is often part of the key scientific information of the data. Therefore, lossless compression algorithms are and will continue to be extremely important in dealing with the distribution of large scientific data sets and in minimizing archival storage requirements. The importance of lossless compression for earth and space science (ESS) and space physics data are addressed by Walker [1].

The First ISLSCP Field Experiment (FIFE) collected a coordinated data set useful for developing and validating models that determine surface climatology from satellite-acquired measurements. Particular attention was placed on measurements of the mass and energy balances at the boundary between the land surface and the atmosphere and on the role of the existing surface biology in controlling the land and atmosphere interactions. The experiment was also designed to explore the use of satellite observations to infer climatologically important land-surface parameters related to the land/atmosphere interactions (see [2] for further details).

The 15 x 15 km FIFE study area included the Konza Prairie Research Natural Area, an NSF Long Term Ecological Research (LTER) site near Manhattan, KS, and surrounding areas. The data collection effort consisted of long-term monitoring through satellite observations and periodic in-situ ground and meteorological measurements along with five intensive field campaigns of 10 to 20 days in length during the growing season of 1987 and late summer in 1989. Attempts were made during the IFC's to obtain coordinated (and at times simultaneous) ground, air, and satellite measurements of hundreds of variables.

Integral to this whole effort was a data system that served as a tool for designing the experiment and for organizing, manipulating, and archiving the complex data set. A dedicated and remotely accessible data system was developed at NASA's Goddard Space Flight Center (GSFC) to meet the data management requirement. Data compression has become a critical concern as the data system prepares the final CD-ROM based archive of the reduced data collection (about 10 Gbytes). To date, one prototype CD-ROM has been produced to evaluate the CD-ROM data publication facilities available at GSFC and to establish operational procedures and software for the final CD-ROM series production (see [3]).

2. FIFE Information System

The fundamental mission of the FIFE Information System (FIS) was to capture, preserve, organize, distribute, and archive the satellite, aircraft, and field data that were collected. During the experiment and to date the image and other large data sets have been stored and distributed on magnetic tape while the smaller data sets have been stored in an on-line data base and distributed via electronic transfer and floppy disk (see [4] for further details). In total, the raw and the derived data products now comprise approximately 120 gigabytes.

The non-image data consist of conventional meteorological and radiosonde data from all NOAA stations within 1° latitude and longitude of the study area, automated meteorological and radiation sensors reporting up to 20 variables every 5 minutes at 32 sites, and in-situ data assembled by the 30 investigator teams during the IFC's. These data sets characterize the diurnal patterns of radiation, moisture and heat flux, atmospheric properties and temperatures, moisture and wind profiles, vegetation and soil condition, and rates of photosynthesis and evaporation. Although overwhelming in diversity, these data comprise less than 1% (300 Mbytes) of the total data volume. The image data on the other hand occupy over 99% of the data volume but consist of only approximately 12 types with differing levels of processing (Table 1).

A final data archive is currently being established. An important element of the archive plan is publication of a set of some 6 CD-ROMs which will contain a reduced data set felt to satisfy the primary objectives of the experiment. The bulk of this data (90^{+}%) will consist of derived (i.e.,

level-1 and level-2) image products from satellite and aircraft based instruments with spatial resolutions from 20 m to 1 km. The need for lossless compression of the image data products arose from the use of the CD-ROMs as the main FIFE data archival product. Dozier and Tilton [5] note that "Since the purpose of the archival process is to keep an accurate and complete record of data, any data compression used in an archival system must be lossless, and protect against propagation of error in the storage media." We feel lossless compression of data stored on CD-ROM media satisfies these criteria quite well.

Table 1: Volume of Selected Digital Image Data (Mbytes)[2]

	Processing Level			
Image Type	Level-0	Level-1	Level-2	Level-3
ASAS	-	3958	-	-
AVHRR-GAC	7406	272	-	-
AVHRR-LAC	67583	1686	812	-
Landsat TM	2157	210	2	-
NS001 (TMS)	19201	1210	24	-
PBMR	-	-	37	38
SPOT-XS	1998	462	-	-
SPOT-PAN	198	144	-	-
TIMS	10775	-	18	-
Totals:	109318	7942	893	38

3. FIS Image Data Products

To be successful, FIFE required image data products that allowed integration of multi-date, multi-sensor, and multi-resolution imagery for quantitative atmospheric and surface biophysical radiometric studies. In particular, mandatory capabilities were to accurately derive, extract, and utilize geographic location and solar and viewing conditions in conjunction with instrument calibration information for derivation of surface radiance and reflectance on a polygonal or per-pixel basis. In FIFE, these outputs were required from the pixel to the regional level as input to models of the surface energy/mass exchange processes.

Based on this set of requirements and the need to distribute the data to a large and diverse user community, a processing approach and operational processing software were developed to handle the task of processing the large volume of level-0 data into complete and user-friendly image products (see [6] for further details). Level-0 image data was defined as unmodified (but possibly reformatted) instrument values as received from the agencies supplying the data. The definitions of the level-0 and higher data products (i.e., level-1, level-2, ...) were derived from and are consistent with the EOS (Earth Observing System) formulated definitions (see [7]).

Of particular concern to the CD-ROM publication and data compression efforts was the content and format of the image products to be published; primarily the level-1 and level-2 products. An example of a generic level-1 FIFE image product is shown in Figure 1. The level-1 products were closely reviewed since they were required to contain all the information necessary to derive at-sensor radiance values and were anticipated to be (and resulted in being) the primary image-data distribution product. Each level-1 product contains a 'header' file consisting of 80 byte ASCII text records that describe the overall contents of the image product, summaries of any necessary calibration, georeferencing, viewing, and solar position information, and comments related to the processing performed. The header file is then followed by a series of files containing the spectral image data, any unpacked and reformatted calibration information, and

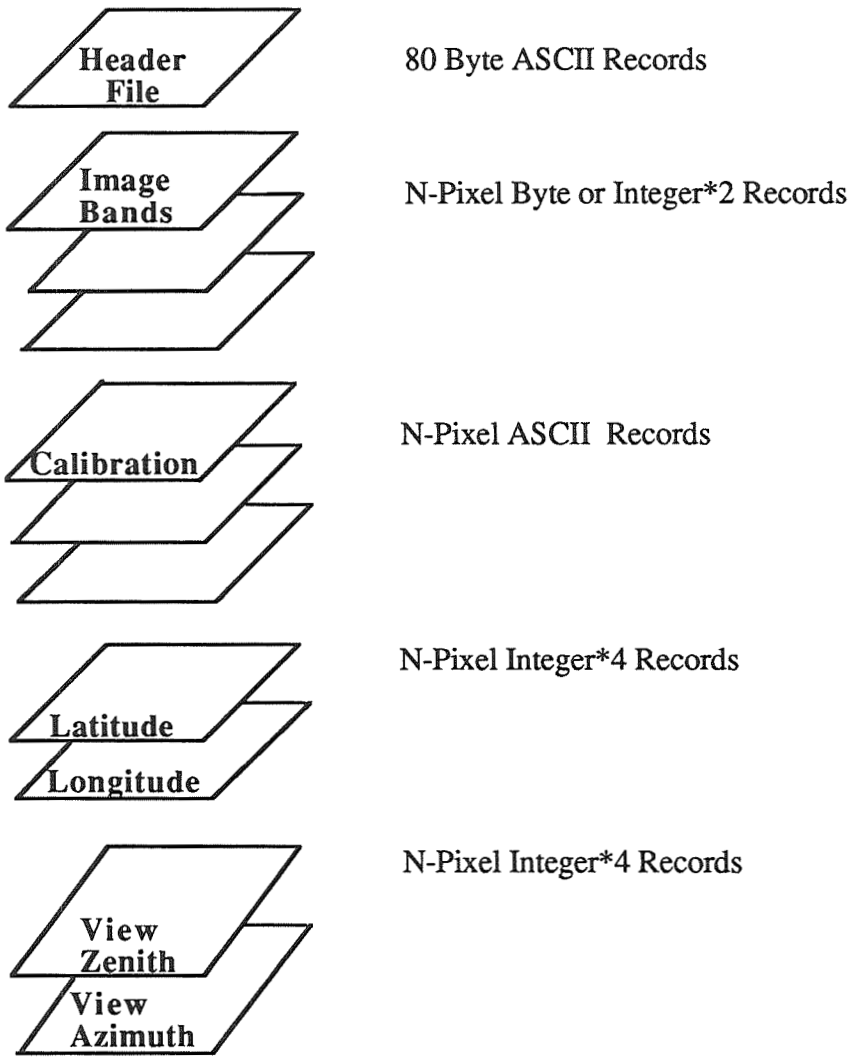
[2]See Appendix A for list of acronyms

georeferencing, view angle, and solar angle information. The georeferencing and view and solar angle files contain values for the respective variables on a per pixel basis for the selected subset of image data. In content, the FIFE level-1 image products are very similar to the level-1 MODIS (Moderate Resolution Imaging Spectrometer) products described by Salomonson [8]. All of these files except for the ASCII header file (which is simply copied) are processed by the data compression software. The ASCII header files were not compressed to allow users to easily review the summary information of a given image and to determine if the image met their particular needs before proceeding with the restoration of the data. In addition, each ASCII header file occupied less than 19K bytes of the total image product volume.

4. Image Data Compression

The need for a lossless compression procedure arose from the desire to store as much image data as possible on the planned CD-ROMs and to preserve the scientific precision of the data. From our experience, it seemed that a useful and intuitive algorithm could be developed to perform the needed compression.

Figure 1: Example Level-1 Standard Data Product



4.1 Algorithm Evolution and Performance

We started with the observation that adjacent pixels in an image frequently have quite similar values (especially in a given line or column). Often too, the range of digital values in an image is only a small fraction of the total data range. Determining an overall image minimum value and subtracting it from all the pixels in the image would therefore maintain the range and distribution of values while bringing the bit representations of the pixel values into the lowest set of bits. In addition, the quantized spectral responses of adjacent pixels (in the form of digital counts or DNs) do not usually differ by more than a few counts. Essentially, the high-order bit layers represent the major structural information of the image and do not change much from one pixel to another. On the other hand, the low-order bits are quite variable between adjacent pixels and can tend to be rather random.

It seemed then that runlength encoding of the high-order bits of the pixels in each line could result in significant compression. Depending on the smoothness of a given image, runlength encoding of the low-order bits may also result in good compression. So, the algorithm was developed to evaluate each series of bit values in a given bit plane across a given image line for runlength encoding. In addition, a threshold was needed to determine whether or not runlength encoding would be worthwhile for a particular line of image data and an alternate storage scheme was needed to process the lines that were not runlength encoded. The decisions on the initial threshold value and what to do with the records that were not runlength encoded resulted from the fact that the first images being considered for compression (level-1 AVHRR-LAC) contained 256 pixels in each line and the runlength counts were being stored in single byte (8 bit) values. If the average runlength for a given bit plane across the pixels in a particular line was greater than 8, then the data would require less space to be stored in a series of 8-bit runlength counts. If the average runlength count was less than or equal to 8, no storage savings could be realized, and the bits were packed into bytes and stored as a series of byte values. Actually, if the average runlength count equalled 8, both encoding formats stored the data equally well, but no storage savings resulted (see Appendix B for details of the original compressed file record structure). This type of bit plane encoding is discussed at some length by Rabbani and Jones [9].

Results of using this first version of the algorithm on a set of 97 FIFE level-1 AVHRR-LAC image products is given in Table 2. The compression ratios for the spectral bands (B1 through B5) ranged from 1.6:1 to 53:1. The values of 16:1 and 53:1 for B1 and B2 were achieved on a night image in which the images for B1 and B2 (which are visible and near infrared bands, respectively) were very smooth with only a small amount of random noise. For the remaining files, the best compression was achieved in the view azimuth (Vaz), view zenith (Vzen), solar azimuth (Saz), and solar zenith (Szen) files. An overall average compression of 2.7:1 was achieved in terms of actual storage space required. This result was promising in that it would allow us to place 270% more AVHRR-LAC imagery on the CD-ROM disc than if we had not compressed the data. The 'bit average' compression ratios in Table 2 (and subsequent tables) are smaller than the 'average' ratios in that they are weighted compression statistics calculated with the actual number of bits in the data rather than the number of bits required for storage. For example, although B1 was stored in a 16 bit value, the actual data only occupied a 10 bit range. Therefore, the bit average ratio of 2.3 for B1 equals the average ratio (3.6) multiplied by the actual bits divided by the storage bits (i.e., 10/16).

The next development in the algorithm resulted from testing the existing software on a new data set, level-1 NS001 TMS (Thematic Mapper Simulator) data. The level-1 NS001 data product consists of 8 spectral bands, 8 associated files of record-by-record housekeeping and calibration information, files of latitude and longitude, and 2 files containing one record each of view azimuth and zenith information for the given flight line. Storage of a nominal level-1 NS001 image product required 8.6 Mbytes for spectral data, 1.5 Mbytes for housekeeping information,

8.6 Mbytes for latitude and longitude information, and 5.5 Kbytes for view angle information. Table 3 shows the results from testing the initial compression scheme on the NS001 data. The highest compression from this original ('old') algorithm was achieved on the latitude, longitude, and view azimuth files. Although the overall compression ratio of 1.5:1 resulted in a space saving of 6.2 Mbytes per image product, we felt we could improve this by more closely evaluating the characteristics of the NS001 and other image data product files.

Table 2: Initial AVHRR-LAC Compression Results

	B1	B2	B3	B4	B5	Anc	Lat	Lon	Vaz	Vzen	Saz	Szen
Storage Bits	16	16	16	16	16	8	32	32	32	32	32	32
Actual Bits	10	10	10	10	10	7	25	26	17	16	17	16
Max Ratio	16	53	2.5	2.8	3.2	1.5	2.2	1.8	6.6	4.0	6.8	6.7
Min Ratio	1.8	1.8	1.6	1.8	1.8	1.4	1.8	1.6	4.8	3.8	4.5	4.6
Avg Ratio	3.6	3.4	1.9	2.3	2.3	1.5	2.1	1.7	5.7	3.8	5.6	5.6
Bit Avg Ratio	2.3	2.1	1.2	1.4	1.4	1.3	1.6	1.4	3.0	1.9	3.0	2.8

Overall Average 2.7:1
Overall Bit Average 1.7:1

Table 3: NS001 Compression Results

	Bands 1 - 8	Housekeeping Files 1 - 8	Lat	Lon	Vaz	Vzen
Storage Bits	8	8	32	32	32	32
Actual Bits	8	7	25	26	17	16
Old Ratio	1.1	0.7	3.4	3.4	4.4	2.4
New Ratio	1.2	1.3	5.5	6.4	1.0	1.0
Old Bit Ratio	1.1	0.7	1.8	3.6	2.2	0.4
New Bit Ratio	1.2	1.1	4.3	5.2	0.5	0.5

Overall Old Ratio 1.5:1
Overall New Ratio 2.1:1
Overall Old Bit Ratio 1.3:1
Overall New Bit Ratio 1.8:1

In the spectral bands the response of the pixels in each column of the NS001 imagery is differentially affected by limb brightening caused by the large across-track scan angle of the instrument. Strong column similarities existed also in the latitude or longitude file (depending on the flight direction) and in the housekeeping files which are series of column formatted ASCII records containing the timing, calibration, and instrument-status information. Corresponding column-element similarities exist also in data collected by linear-array (push-broom) scanners. Based on this, it was decided to take advantage of these column similarities by determining the minimum value of each data column, storing these values as part of the compressed file structure, and subtracting the minimum column value from each element in the column. This resulted in a more 'column normalized' data matrix and the subtraction of the overall image minimum was no longer needed.

The values in each record of the georeferencing, view-angle, and solar-angle files for the image products were also quite similar depending on the orientation of the image. Although not applicable to the NS001 data, other remote sensing instruments (like Landsat TM) gather 'n' lines

of data in each across-track scanning pass with a set of n detectors. Since each detector has a somewhat different response, consecutive lines of data over the same target can contain somewhat different values. It was assumed that these line-to-line differences would be somewhat 'independent' of the column similarities and would still exist in the 'column normalized' data matrix. Determining, storing, and subtracting the row minimum values was used to remove these row similarities and to bring the data down into a 'smallest' number of bits for subsequent encoding.

The results of applying this refined algorithm on the NS001 data are given as the 'new' values in Table 3. Although only incremental improvements were realized in compressing the spectral-image files, and the view-azimuth and view-zenith files remained at their original volumes, an overall improvement in compression of 20% was realized for the whole image product. The decrease in the compression of the view angle files is due to the fact that each file contains only one record of information and the additional 'overhead' storage required to store the column minimum values which was not present in the old version. Since these files comprise less than 0.03% of the data volume, we felt this was a small price to pay for maintaining the generality of the compression overall. If the files contained a record for each image line as did the level-1 AVHRR-LAC data, we believe they would compress at least as well as the view angle files in the AVHRR-LAC data. As is evident from the values in Table 3, the real savings came from the housekeeping and latitude and longitude file compression which improved by 66% and 12%, respectively. The small improvement in the spectral data compression and the large improvement in compressing the latitude, longitude, and housekeeping files (which comprised a significant part of the data product volume) resulted in the 20% improvement. Overall, a 2.1:1 compression was realized in relation to actual data storage space requirements.

After testing the improved algorithm on the NS001 data, we went back to check it on the level-1 AVHRR-LAC data. Table 4 contains the results of applying the updated algorithm to the 707 level-1 AVHRR-LAC data products which were processed for CD-ROM publication. The largest improvements in the compression again came from the view angle, solar angle, and latitude and longitude files. Overall an additional 8% improvement in storage space was realized. Although this seems small, it resulted in saving an additional 40 Mbytes (about 6.7% of the total CD-ROM volume) in storing the level-1 AVHRR-LAC data. This allowed all the level-1 satellite monitoring data (i.e., AVHRR-LAC, Landsat TM, and SPOT) from 1987 through 1989 to be archived on a one rather than two CD-ROMs.

Table 4: Updated AVHRR-LAC Compression Results

	B1	B2	B3	B4	B5	Anc	Lat	Lon	Vaz	Vzen	Saz	Szen
Storage Bits	16	16	16	16	16	8	32	32	32	32	32	32
Actual Bits	10	10	10	10	10	7	25	26	17	16	17	16
Max Ratio	64	85	2.7	3.1	3.1	1.5	4.4	8.4	43	102	22	24
Min Ratio	1.7	1.7	1.4	1.3	1.3	1.5	2.3	2.0	6.8	7.6	6.0	4.6
Avg Ratio	3.6	3.4	1.9	2.3	2.3	1.5	3.1	2.4	8.6	33	8.5	18
Bit Avg Ratio	2.3	2.1	1.2	1.4	1.4	1.3	2.4	2.0	4.6	16	4.5	9.1

Overall Average 3.5:1
Overall Bit Average 2.2:1

Tables 5, 6, and 7 contain the results of applying the algorithm to Landsat Thematic Mapper (TM), SPOT multispectral (XS), and SPOT panchromatic (PAN) 'browse' image products. Note that the actual level-1 image products could not be distributed without infringing on EOSAT or CNES copyrights. Therefore, the data were spatially degraded by averaging the original pixels

within a 2 by 2 moving window. The resultant degraded Landsat TM, SPOT-XS, and SPOT-PAN images were termed as 'browse' products with spatial resolutions of 60, 40, and 20 meters, respectively. Compared to the original level-1 imagery, these would be considered to be lossy data products. The largest compression ratios were achieved in the latitude files, with the longitude files also compressing significantly. The spectral bands themselves were compressed an average of 1.4:1. This is lower than the AVHRR-LAC results but higher than the NS001 spectral data compression. The increase over the NS001 data which is of similar spatial resolution and spectral content is likely due to the averaging done in degrading the original data to derive the browse products.

Table 5: Landsat TM Compression Results

	B1	B2	B3	B4	B5	B6	B7	Lat	Lon
Storage Bits	8	8	8	8	8	8	8	32	32
Actual Bits	8	8	8	8	8	8	8	25	26
Max Ratio	1.5	1.8	1.5	1.3	1.1	1.7	1.3	16	18
Min Ratio	1.3	1.5	1.2	1.1	1.0	1.2	1.1	15	5.9
Avg Ratio	1.4	1.6	1.3	1.1	1.0	1.4	1.2	16	7.3
Bit Avg Ratio	1.4	1.6	1.3	1.1	1.0	1.4	1.2	13	5.9

Overall Avg 2.4:1
Overall Bit Avg 2.1:1

Table 6: SPOT-XS Compression Results

	B1	B2	B3	Lat	Lon
Storage Bits	8	8	8	32	32
Actual Bits	8	8	8	25	26
Max Ratio	1.9	1.9	1.5	21	19
Min Ratio	1.0	1.0	1.1	5.2	3.8
Avg Ratio	1.5	1.5	1.2	15	8.6
Bit Avg Ratio	1.5	1.5	1.2	7.0	7.0

Overall Avg Ratio 3.8:1
Overall Bit Ratio 2.8:1

Table 7: SPOT-PAN Compression Results

	B1	Lat	Lon
Storage Bits	8	32	32
Actual Bits	8	25	26
Max Ratio	1.5	20	20
Min Ratio	1.3	17	6.5
Avg Ratio	1.4	18	9.7
Bit Avg Ratio	1.4	14	7.9

Overall Avg 6.7:1
Overall Bit Avg 5.5:1

Preliminary tests of the algorithm on level-1 ASAS and level-2 AVHRR-LAC image products, have resulted in average compression ratios of 2.5:1 and 2.2:1, respectively. The level-1 ASAS image product consists of seven spectral data files (one for each of seven look angles) containing 29 spectral bands and an ASCII header file. The spectral image bands are sequentially extracted from the look angle files and compressed. The level-2 AVHRR-LAC product contains an ASCII header file, two files containing cloud and urban/water feature masks, two files of reflectance values, one file of surface temperature values, one file of vegetation index values, and five files of original spectral data. Each of the files has been resampled to a spatial resolution of approximately 300 meters from the original 1 km resolution. The better compression of the ASAS data over that of NS001 is due to the increased similarity of pixels from its narrow spectral bands and the linear arrays of detectors that gather data. Both of these features would lend themselves to enhanced compression based on the steps in the algorithm. The level-2 AVHRR-LAC results are overall comparable to the compression results for the spectral bands shown previously in Table 4.

4.2 Algorithm Implementation

The ordered steps of the improved operational algorithm include : 1) normalization of the data matrix columns by subtracting the respective column minimum value from each element in the column, 2) normalization of the rows by subtracting the respective row minimum value from each value in the row, and 3) viewing the remaining values in each row as layers of bits that are either runlength encoded or packed back into bytes depending on the calculated average runlength of the line. A description of the improved compressed file format is provided in Appendix C.

The compression package, which consists of a base set of compression functions for handling 8, 16, and 32-bit values and processing control functions, is written in C and currently runs on a VAX computer in the Laboratory for Terrestrial Physics at NASA GSFC. Adherence to modularity of the sourcecode functions has resulted in the ability to implement compression of a new image-data product in a day or less. A complementary set of image-restoration software, also written in C, was developed, and can be run on several computing platforms including IBM compatible PCs, Macintosh (Plus, SE, and II), Sun and SPARC workstations, Personal Iris, and HP 9000 RISC workstations. This system independence was required to allow use of the data from the distributed CD-ROMs by as large a group of science users as possible. Although total system independence is not achieved, implementation of the restoration software on any of the mentioned machines simply requires editing a single line of the C source code file to set the type of system architecture (i.e., Intel or Motorola) being used before compiling and linking the package.

An average time of 12 minutes is currently required to compress a FIS level-1 AVHRR-LAC image product consisting of 13 files from 2.4 Mbytes to an average of 0.7 Mbytes on a VAX 11/780 system. A more modern system such as the VAX 6000 planned for use in the Boreal Ecosystem Atmosphere Study (BOREAS) (see [10]) could reduce this time to 1 minute or less. Restoration of the image product requires an average of 10 minutes on the loaded VAX 11/780 system; an average of 8 minutes on a 286 based PC; and 8 seconds on a Hewlett-Packard Series 9000 Model 720 RISC workstation.

5. Conclusions

The overall compression ratios realized by using the described algorithm on the FIFE image-data products are consistent with the rule-of-thumb limit of 2:1 for lossless image compression algorithms. The algorithm will definitely serve its purpose for the CD-ROM production effort in decreasing the amount of storage required for distributing the image data and has shed some

light on its potential use and enhancement for future efforts. For each image product, an overall ratio of 2:1 or greater was achieved from the compression, which allows production of only half as many CD-ROMs as would have otherwise been required. Although the original intent was to compress the spectral data, the best compression ratios were achieved in the georeferencing and view and solar-angle files. The compression of the latitude and longitude files within each of the AVHRR-LAC and NS001 products was quite similar, but there was a notable difference in the compression of the latitude and longitude files in the Landsat TM and two SPOT image products. In these three, the compression ratio of the latitude files were generally twice as large as the compression for the longitude files. Review of these files showed that the least-squares equations used in calculating the latitude coordinates were predominantly linear in nature while the equations derived to calculate the longitude coordinates were generally quadratic. For the one or two instances where the longitude coordinate equation was linear, the longitude coordinate file compressed comparably with the latitude coordinate file. This nonlinearity is the result of the small second-order effect caused by the convergence of the lines of longitude over the scenes, which is not present in the latitude values. This effect will be more important in northern latitudes, such as the proposed BOREAS site.

For current purposes, the algorithm is 'frozen' and operational in producing the FIFE CD-ROM image data sets, however, we are exploring options for improving on the current results. One possible improvement in the compression would be to map the remaining values in each line (after column and row minimum removal) into a Gray code (as discussed by Rabbani and Jones [9]) before proceeding with the runlength encoding. In theory, this should improve (increase) the length of bit runs for runlength encoding purposes. These improvements would benefit the upcoming BOREAS project, in which CD-ROM is being considered as a primary data-distribution mechanism for the satellite image data. It would also benefit distribution of the data using available communications networks.

The algorithm's performance is quite broad (as are most compression algorithms) depending on the data layer being processed, but its generality has been a significant point in handling the FIFE data sets. Based on our experience with the algorithm, we envision this 'intuitive' compression algorithm to be useful on a broad range of spatial data sets including gridded layered modeling data sets which could contain terrain, spectral, and meteorological variables. Such data sets will be required for coordinated earth systems field experiments executed during the next decade.

Acknowledgements

The work reported here was supported by the FIFE project through money supplied by the Land Processes Branch, NASA Headquarters, under contract to Hughes STX Corporation and VERSAR Incorporated. The authors thank Andy Newton (University College London, England) for his feedback on the initial algorithm implementation which made us further examine portability of the image restoration software and resulted in a much more useable package.

References

[1] Walker, R.J., "Scientific Requirements for Space Science Data Systems", *Proceedings of the Space and Earth Science Data Compression Workshop*, NASA Conference Publication 3130, pp. 37 - 50, April 11, 1991.

[2] Sellers, P.J., F.G. Hall, G. Asrar, D.E. Strebel, and R.E. Murphy, "The First ISLSCP Field Experiment (FIFE), *Bulletin of the American Meteorological Society*, vol. 69, no. 1, pp.22-27, 1988.

[3] Strebel, D.E., D. Landis, J.A. Newcomer, J. DeCampo, and B.W. Meeson, "Collected Data of the First ISLSCP Field Experiment, Prototype Volume", Published on CD-ROM by NASA, 1991.

[4] Strebel, D.E., J.A. Newcomer, J.P. Ormsby, F.G. Hall, and P.J. Sellers, "The FIFE Information System", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 28, no. 4, pp. 703-710, July 1990.

[5] Dozier, J. and J.C. Tilton, "Data Compression for Data Archival, Browse, or Quick-Look", *Proceedings of the Space and Earth Science Data Compression Workshop*, NASA Conference Publication 3130, pp. 61 - 65, April 11, 1991.

[6] Newcomer, J.A., S.J. Goetz, D.E. Strebel, and F.G. Hall, "Image Processing Software for Providing Radiometric Inputs to Land Surface Climatology Models", *Proceedings of IGARSS'89*, vol. 3, pp. 1779-1782, 1989.

[7] NASA, "Report of the Eos Data Panel", Volume IIa, NASA Technical Memorandum 87777, 1986.

[8] Salomonson, V.V., "The Moderate Resolution Imaging Spectrometer: An Eos Facility Instrument Candidate for Applications of Data Compression Methods", *Proceedings of the Space and Earth Science Data Compression Workshop*, NASA Conference Publication 3130, pp. 13 - 23, April 11, 1991.

[9] Rabbani, M. and P. Jones, *Digital Image Compression Techniques*, SPIE Optical Engineering Press, Volume TT7, 221 pages, 1991

[10] Sellers, P., J. Cihlar, M. Apps, B. Goodison, F. Hall, B. Harriss, D. Leckie, E. Ledrew, P. Matson, and S. Running, *Charting the Boreal Forest's Role in Global Change*, Eos, Vol. 72, No. 4, pp. 33, 35, and 40, January 22, 1991.

Appendix A
FIFE Image Product Acronyms

ASAS	-	Advanced Solid-State Array Spectroradiometer
AVHRR-LAC	-	Advanced Very High Resolution Radiometer - Local Area Coverage
AVHRR-GAC	-	AVHRR - Global Area Coverage
Landsat TM	-	Landsat Thematic Mapper
NS001 (TMS)	-	NS001 Thematic Mapper Simulator
PBMR	-	Push-Broom Microwave Radiometer
SPOT-XS	-	Systeme Probatoire d'Observation de la Terre (Multispectral mode)
SPOT-PAN	-	SPOT (Panchromatic mode)
TIMS	-	Thermal Imaging Multispectral Scanner

Appendix B
Original Compressed File Format

[Note that two-byte fields are in VAX byte order (i.e., low-order byte first)]

Record 1 (Header record)

- Bytes 1 - 4 : Overall image minimum value
- Byte 5 : Number of bits being encoded for each image line (NBits)
- Byte 6 : Total number of bits in the file values (8, 16, or 32)
- Byte 7 : Number of 256 element segments across a line of the image (NSeg)
- Bytes 8 - 9 : Number of records (lines) in the original file (NLines)

Records 2 - n (Encoded data records) (where n = NBits * NSeg *NLines + 1)

Runlength Encoded Record

- Byte 1 : Record type (0 = runlength encoded)
- Byte 2 : First runlength bit value (equal 0 or 1)
- Byte 3 : Number of runlength counts that follow (m)
- Byte 4 : Runlength count - 1 for first bit value (record byte 2)
- Byte 5 : Runlength count - 1 for opposite bit value

.
.

- Byte m+3 : Runlength count - 1 for last bit series

Bit Encoded Records

- Byte 1 : Record type (1 = bit encoded)
- Byte 2 : Contains packed bits for segment values 1 - 8
- Byte 3 : Contains packed bits for segment values 9 - 16

.
.

- Byte 32 : Contains packed bits for segment values 249 - 256

Appendix C
Improved Compressed File Format

[Note that two and four-byte fields are in VAX byte order (i.e., low-order byte first)]

Record 1 (Header record)

Byte 1 : Total number of bits in the file values (8, 16, or 32) (Total_Bits)
Bytes 2 - 3 : Number of lines/records in the original file (NLines)
Bytes 4 - 5 : Number of values per line (NVals)

Record 2 (Column minimum record)

Bytes 1 - n : Column minimum values (1, 2, or 4 bytes each depending on Total_Bits) (NVals in number)

Records 3 - z (Encoded sets of compressed data information)

Each set of information used to reconstruct an output data file record contains a Row_Minimum value, an NBits value, and a set of encoded records in the following manner:

Row_Minimum : Stored as an 8, 16, or 32 bit value depending on the respective value of Total_Bits in the header record.
NBits : Stored as an 8 bit value. Indicates the number of bits that were left for encoding in this line after column and row minimum subtractions.

Series of NBits encoded data records as Runlength encoded or Bit encoded records

Runlength Encoded Record

Byte 1 : Record type (0 = runlength encoded)
Byte 2 : First runlength bit value (equal 0 or 1)
Byte 3 - 4 : Number of runlength counts that follow (Nr)
Byte 5 - 6 : Runlength count - 1 for first bit value (record byte 2)
Byte 5 : Runlength count - 1 for opposite bit value

.
.

Byte m - m+1 : Runlength count for last bit series, where m = 2*Nr + 3

Bit Encoded Records

Byte 1 : Record type (1 = bit encoded)
Byte 2 : Contains packed bits for segment values 1 - 8
Byte 3 : Contains packed bits for segment values 9 - 16

.
.

Byte J+1 : Contains packed bits for last values in the record where
J = NVals/8 if NVals is a multiple of 8 or
J = NVals/8 + 1 if NVals is not a multiple of 8

THE EFFECTS OF VIDEO COMPRESSION ON ACCEPTABILITY OF IMAGES FOR MONITORING LIFE SCIENCES' EXPERIMENTS

Richard F. Haines and Sherry L. Chuang
Spacecraft Data Systems Research Branch
Ames Research Center - NASA
Moffett Field, CA 94035

1. Introduction

Current plans indicate that there will be a large number of life science experiments carried out during the thirty year-long mission of the Biological Flight Research Laboratory (BFRL) on board Space Station Freedom (SSF) [1]. Non-human life science experiments will be performed in the BFRL. Two distinct types of activities have already been identified for this facility: (1) collect, store, distribute, analyze and manage engineering and science data from the Habitats, Glovebox and Centrifuge, (2) perform a broad range of remote science activities in the Glovebox and Habitat chambers in conjunction with the remotely located principal investigator (PI). These activities require extensive video coverage, viewing and/or recording and distribution to video displays on board SSF and to the ground. This paper concentrates mainly on the second type of activity. Each of the two BFRL habitat racks are designed to be configurable for either six rodent habitats per rack, four plant habitats per rack, or a combination of the above. Two video cameras will be installed in each habitat with a spare attachment for a third camera when needed. Therefore, a video system that can accommodate up to 12-18 camera inputs per habitat rack must be considered.

The present Glovebox (GB) design provides an enclosed, bioisolated workspace in which a wide variety of non-human life sciences research can be conducted without contaminating the rest of the interior of SSF. Two dedicated video cameras are installed in the walls of the GB to provide orthogonal views of the activity in the work volume. These cameras are in addition to others installed in the attached Modular Habitats. The user will be able to select and display video data from the work volume, GB attached habitats, transmissions from the ground or other external video sources.

The centrifuge provides a selectable gravity environment (between .01g and 2g) for biological specimens. In addition, four rodent habitats can be accommodated on an inner ring of the centrifuge. Each of these habitats will also have two video cameras installed with a spare attachment for a third camera when needed. Therefore, from 24-36 camera outputs must be supported in the centrifuge system.

2. Problem and Approach

The number of individual cameras in the entire BFRL can range between 50-74 cameras (when all of the components of the BFRL are in use), each simultaneously collecting NTSC quality image data. In addition, there will be a limited transmission bandwidth that will (likely) be available between SSF and the ground. An approach to these problems may be through the use of video compression technologies.

3. Methodology and Results

In general, the research approach described elsewhere [3] was used here. Video images were viewed by a variety of Ss consisting of PIs and other professional personnel already familiar with the scene content. Each made judgments of the quality and acceptability of each compressed scene. The data were then subjected to statistical analyses in order to be able to extend the parametric findings beyond the present groups of participants. Three independent experiments were conducted, one involving static images of plants and two involving moving animals. Each is described in detail below.

3.1 Experiment 1: Still Image Compression of Three Plant Scenes

3.1.1 Procedure and Test Instructions. The subject (S) was told about the nature of the study, what he or she was supposed to do, and shown the apparatus and scoring sheet and rating criteria sheet which were posted nearby. There were three scenes per experiment and three primary subjective judgments made per scene. In the first experiment there were two additional judgments made on each scene. The method of pair-comparison [2] was employed whereby each of the four levels of image compression were presented in all possible pairs. These twelve image pairs were displayed side-by-side on a high resolution color monitor with the highest resolution image located on the right or left side of the screen on a random basis. Each image measured 6.75" x 6.75" (45.5 square inches).

The *first* subjective judgment required was to carefully inspect both images on the screen and select the one having the best overall quality to support the S in carrying out scientific research. The *second* subjective judgment to be made [only with respect to the screen image chosen (above) as having the best overall quality] was a numeric rating from 1 to 5 where: 1 = completely unacceptable image quality, 3 = average image quality, and 5 = maximally clear and acceptable image quality. The *third* subjective judgment required was to either accept or not accept the image chosen in terms of whether it would be acceptable to provide the kinds of answers to questions the S would normally ask of this particular image. The *fourth* subjective judgment to be made was to list which image characteristics were used in making the numeric rating of image quality such as color, contrast, brightness, resolution, etc. The *fifth* requirement was to circle those parts of the image (on a B&W copy of the screen image placed in front of the S) at which the S looked in order to make his or her judgments. This was done on a trial by trial basis for later analysis.

3.1.2. Apparatus. The apparatus consisted either of a Panasonic color CCD (model WV-CD 110A) camera (for scene 1), or a Toshiba color CCD (model IK-M30) microminiature camera yielding over 360 TV lines horizontally (for scenes 2 and 3) taken through a microscope. An image capture board ("Moonraker" by Workstation Technology Inc.) and Joint Photographic Expert Group (JPEG) standard compression board ("Picture Press" by Storm Technology, Inc.) were installed in a Macintosh II with 8 megabytes of RAM and a 1.04 Gigabyte hard disk. Software image control was accomplished using image manipulation software ("Photoshop" by Adobe, Inc.). The images were displayed on a 20 inch (diagonal), high resolution (1024 x 768 pixel) color monitor (Mirror Technologies, Inc.; model C/T 20HA, Rev. G).

3.1.3 Image Compression Details. Following are the four image compression levels investigated using the JPEG standard. The number by each level is referred to in the results section. 1 = Excellent; 2 = High; 3 = Good; and 4 = Fair. Table 1 presents selected information on the four levels of compression derived from the scenes which were studied here.

Table 1. Compression Details Associated With Test Scenes Studied Here

	Scene 1 "wheat stalk"				Scene 2 "wheat kernel cluster"				Scene 3 "magnified single wheat kernel"			
comp. level	Excel	High	Good	Fair	Excel	High	Good	Fair	Excel	High	Good	Fair
size												
bits/pix.												
comp. ratio												
comp. time (sec.)	1.60	0.80	0.78	0.60	1.58	0.80	0.78	0.78	1.55	0.90	0.78	0.76

Notes:
Excellent settings uses 1:1:1 subsampling ratio
The range of compression ratios is between 5:1 to 7:1
High setting uses 2:1:1 subsampling ratio
The range of compression ratios is between 16:1 to 24:1
Good setting uses 2:1:1 subsampling ratio
The range of compression ratios is between 40:1 to 60:1
Fair setting uses 2:1:1 subsampling ratio
The range of compression ratios is between 80:1 to 120:1

The S sat with his or her eyes 32" (+/- 2") from the screen of the monitor so that the angular width of the two images subtended approximately 20 degrees arc. Each of the three compressed test scenes was presented with each of the others in all combinations in random order for a total of twelve pairs per S per scene. The order in which the three scenes was presented also was randomized.

3.1.4 Test Scenes. Three static color images were investigated: (1) a "wheat stalk" cluster scene, (2) "wheat kernel cluster" scene and, (3) a magnified "single wheat kernel."

3.1.5 Test Subjects. A total of ten volunteer Ss took part. Most were senior level NASA investigators, contractors, or visiting faculty working in such fields as plant physiology and biology, closed environment life support research and development, plant nutrition. Three were graduate students working at Ames in the area of plant growth dynamics for the SSF program.

3.1.6 Results. The results are presented in four sections, each of which deals with the subjective judgments that were made on each compressed image.

Image Accuracy Judgement. As expected, (1) the larger the difference in compression levels between the two paired images the greater was the accuracy of judgement. (2) Of the three pair-comparison conditions that were one compression level apart (viz., 1,2; 2,3; 3,4), only 28 percent were judged accurately. (3) Of the two pair-comparison conditions that were two compression levels apart (viz., 1,3 and 2,4), 42 percent were reliably correct. (4) Of the single pair-comparison that were three compression levels apart (viz., 1,4), 67 percent were reliably

correct. (5) There were differences in the proportion of unreliable judgements due to the kind of scene presented, and (6) There was only a relatively small difference in these judgements due to whether the best image was located on the left or the right side of the screen as would be expected. Nine of the eighteen cells (50 percent) were only ten percent apart indicating a generally good level of agreement. Five of the 18 cells (28 percent) were twenty percent apart, two (11 percent) were thirty percent apart, one (5 percent) was forty percent apart, and one (5 percent) was identical. The single cell in which there was perfect agreement was at the 50 percent "guessing" level and is, therefore, not considered reliable.

Image Numeric Ratings. There were no statistically significant differences (by analysis of variance) among the four JPEG compression levels. The three scenes tested did produce a significant difference ($F = 8.25$; $df=2$; $p=0.009$) with scene 2 producing the largest subjective quality ratings for each compression level (mean = 3.70).

Image Acceptability Judgement. There were small (statistically insignificant) differences among these means. An analysis of variance showed that the four JPEG compression level means were not significantly different from one another but that the three scenes were ($F = 9.73$; $df = 2$; $p = 0.006$). These mean acceptability results parallel (in magnitude) the above mean image rating results across these four compression levels and three scenes.

Image Characteristics Selection Results. Each S was asked to indicate which particular image characteristics were used in making a judgment. It was found that resolution by itself was the single-most important image characteristic regardless of scene content. This was followed in frequency of occurrence by resolution, color and brightness/contrast combined.

3.1.7 Conclusions. These results suggest that there is no clearly perceptible difference in the ratings of image quality among any of the four JPEG compression levels studied here for any of the three scenes presented. There was a significant difference found due to the type of scene studied. The magnified image of scene 3 was significantly darker than were the other two and was difficult to discern the identity of specific tissues. It elicited the lowest mean rating of all three scenes across these four JPEG compression levels. For scenes that are clearly familiar to the viewer and possess sufficient resolution, brightness, and contrast a FAIR JPEG compression level (i.e., average 10 kilobytes/image) appears to be sufficient.

3.2 Experiment 2: Motion Image Compression of Three Rodent Scenes

3.2.1 Procedure and Test Instructions. Each S viewed fifteen second-long dynamic scenes four times, each generated at a different level of video compression. First they had to assign the scene a number from one to five indicating its image quality where 1 = completely unclear and unacceptable image quality, 3 = average image quality, and 5 = maximally clear and acceptable image quality. Their second judgment was to answer yes or no to the question of whether that scene would be acceptable to them in order for them to answer the kinds of questions they would normally ask of this particular image. Finally, using the image characteristic key that is posted to their left[1], to write down which specific image details led them to select the numeric rating (1 to 5) they chose."

Three rodent scenes were selected. Scene 1 consisted of general animal movement including a (**Jump**) . Scene 2 showed both rodents in **Play** activity where they rolled on top of each other and chased each another around the enclosure. The third scene was of a subtle "fall-over" behavior which was of interest to several Ss. In this **Fall-Over** scene both animals were generally sedentary.

[1] This sheet was located about 26" away at eye level.

3.2.2 Apparatus. The original rodent behavior imagery was recorded on a NASA-Ames' animal centrifuge using a Panasonic CCD (model WV-CD-110A) camera with 16mm fixed focus lens located outside the transparent animal holding cage. This signal was routed to a BetaCam recorder through a slip-ring assembly on the rotational axis of the centrifuge and thence to a SVHS recorder.

Scenes selected from the SVHS tape were compressed to four levels (384, 448, 768, 1540 kbits/sec.) using a *Compression Labs Inc.'s* "Rembrandt" model Codec and then re-recorded on a new SVHS tape in random order. The S never knew which compression level was being shown nor was he or she shown NTSC broadcast quality imagery prior to testing. The display apparatus consisted of SVHS color tapes (only) on which the compressed images were recorded, a SVHS tape recorder for playback, and a NEC (model PR-2000S) color TV monitor with 20" (diagonal) screen.The S sat with his or her eyes normal to the TV screen and 32" away.

3.2.3 Test Subjects. Twelve people took part as volunteer subjects. Eight were senior level staff who regularly worked with rodents in such disciplines as physiology, neuromuscular dynamics, behavioral interactions, etc. Four people were animal care technicians who were very familiar with the health and status characteristics of white rats.

3.2.4 Results. The results are presented in three sections, *image rating, image acceptability, image characteristics.*

Image Rating Results. The differences in mean image ratings related to variation of compression level were not statistically significant. However, as expected, the mean ratings among the three scenes waeresignificantly different ($F = 5.01$; $df = 2$; $p = 0.0025$). The mean ratings for scene 3 "fall over" approached significance ($p=0.06$). As was found in Experiment 1, the type of experimental situation that must be compressed/decompressed and visually analyzed plays a very significant role in the image rating.

Image Acceptability Results. The percentage of the twelve Ss who selected each scene as being acceptable tended to increase as bandwidth increased for all three scenes. These mean data suggest that the largest gain in image acceptance is between 448 and 768 kilobits per second.

Image Characteristics Selection Results. Resolution, motion, and brightness/contrast (combined) are the most frequently selected image characteristics across all three scenes. In addition color, by itself, was *never* selected as being an important characteristic of these images.

3.2.5 Conclusions. Experiment 2 has shown that these four compression levels did not yield a statistically significant difference in mean image ratings while the mean ratings for the three test scenes were significantly different. The largest difference in image acceptability across these four bandwidths occurred between 448 and 768 kilobits per second, averaging 36 percent increase in mean acceptance.

3.3 Experiment 3: Motion Image Compression of Three Primate Scenes

3.3.1 Procedure and Test Instructions. The same procedures and instructions were used as just described for Experiment 2 except that, (a) a mature squirrel monkey was the animal of interest, and (b) 576 kilobits/sec was substituted for 448 kilobits/second in order to provide an additional data point on possible curve-plots of the joint results. The three test scenes selected were as follows: Scene 1 showed a relatively close-up view with the entire face of the monkey filling the field of view. This scene is referred to as **Face**.

The second scene, **Rear of Head,** showed the rear of the animal's head with a hard plastic electrode cap attached and the restraining device in which the animal sat. Of particular

experimental monitoring interest by the Ss was the color of the skin surrounding the cap, fur condition and color, and other apparatus detail. The third scene was a close-up of the eyes - it is referred to as **Eye Close-up**.

3.3.2 Apparatus. The hardware was identical to that used in Experiment 2 except that the CLI compression device was pre-set to 384, 576, 768, and 1,540 kilobits per second for the SVHS video tape made.

3.3.3 Test Subjects. Eleven volunteers took part (9 males, 2 females). All were NASA personnel or contractors working with these animals routinely and all were different from the Ss used in the first two studies.

3.3.4 Results. The results are presented in three sections, *image ratings, image acceptability, and image characteristics.*.

Image Rating Results. Separate analyses of variance were conducted on the data from each scene. The means of the four bandwidths were significantly different for each of the three scenes: Scene 1 ($F = 9.36$; $df = 3$; $p = 0.0001$); Scene 2 ($F = 4.88$; $df = 3$; $p = 0.0055$); and Scene 3 ($F = 4.35$; $df = 3$; $p = 0.0096$). Another ANOVA incorporating all of the data found that the scene main effect was also highly significant ($F = 11.1$; $df = 2$); $p = 0.0001$). While the bandwidth main effect in this analysis was statistically significant it was not as large an effect as found in the individual scene analyses (all $p < 0.01$).

Image Acceptability Results. The percentage of the eleven subjects who selected each image as being acceptable increased regularly with increasing bandwidth as expected.

Image Characteristics Selection Results. The image characteristics which were cited as being important to each participant's judgments were rank ordered by frequency of occurrence. Resolution and color were considered to be the most important image characteristics across all three scenes. Neither color nor brightness/contrast was ever selected (except once) as being an important image characteristic.

3.3.5 Conclusions. Perhaps the most general conclusion that can be drawn from Experiment 3 is that for each of the three scenes selected, this range of bandwidths produced a clearly significant increase in judged image quality. The largest increase in image acceptability occurred between 384 and 576 kilobits per second, increasing from 45 to 79 percent (mean) acceptance level.

4. Discussion: General Overview of Findings.

The degree to which a still or moving image can be compressed and decompressed and remain acceptable and useful depends upon numerous hardware, software, and humanware "factors". With regard to motion image compression, the present study has shown that for the four dynamic (inter-frame compression) bandwidths examined here using a proprietary algorithm, mean image ratings increase reliably as compression level decreases. The results from the second experiment involving rodents may be compared directly with previous data (Haines and Jackson, 1990; fig. 1a) in which the same codec and compression levels were used as well as white rats within enclosures. When the slight difference in rating scale indices used in these two studies is taken into account it is seen that the mean ratings are almost identical. For example, the 384, 448, and 768 kilobits per second compression levels result in a slightly above average mean ratings of 3.5 while the 1,540 kilobits per second compression level yields a full point higher.

The Ss' recommendation on the type of scene that was evaluated also played a significant role in judged image quality. In addition to the selection of scenes based on representative experiments, scenes were purposefully selected which would sample all of the basic visual perception domains involved in anticipated future Space Station Freedom life science operations. Domains such as high and low visual resolution were included as well as a wide range of colors, brightness, and contrasts.

It is true that future Space Station Freedom life science procedures will probably differ from the rather passive animal monitoring and plant examinations carried out here. Nevertheless, if PIs must visually inspect in-space specimens from the ground using video compression hardware and software, the fundamental image features they will look at will be similar to those studied here. The kinds of specimens also may be different but the range of sizes of critical detail to be inspected cannot be much greater (without the use of high definition TV). Likewise, unless digital image processing techniques are used that involve pseudo-coloring, edge enhancement, etc., the range of image brightness and contrast cannot be much greater than what was presented here. Of course this is also true for other types of TV sensors such as low light or infra-red since it is the final display which the S looks at which determines the ultimate image contrast and brightness. It is for these reasons that the presented results are reasonably representative of those that would be found if other life science specimens (which possess different dynamic behavior) were substituted.

5. Conclusions

The JPEG standard was found to provide acceptable still-frame imagery of plants at compressions as high as 120:1, depending upon particular scene content. Resolution by itself was most important for the still-frame imagery followed by resolution combined with color or brightness/contrast. For moving imagery using an inverse cosine transform compression algorithm, a transmission bandwidth of about 768 kb/s or one-half of T-1 was found to provide very high acceptability for the mean of the three scenes in which camera imagery was colorful and showed high detail. Finally, the visual judgment criteria that were selected most often as being important for evaluating dynamic imagery was resolution, color, and image motion in some combination. The present testing methodology which involved individual Ss evaluating their own data was effective in evaluating video compression effects.

A wide array of local (in space) and remote (on the ground) visual judgments will be made on plant and animal specimens on board Space Station Freedom in the future. While this and other studies have shown that carefully selected video compression techniques provide an acceptable solution to transmission bandwidth constraints, the final quality of the remote television imagery that is achieved will be dependent upon complex, interrelated hardware, software (video architecture), and humanware factors. Advanced pre-flight simulations using representative flight end-to-end hardware should be conducted in order to optimize this imagery and related scientific procedures, particularly the role that can be played by infra-red imagery, switching and scheduling algorithms in order to optimize the use of available transmission bandwidths.

References

[1] *Space Exploration's Centrifuge Facility Project Communications Plan*, Communications and Network Development Branch, Ames Research Center, NASA, (Draft 2/27/90) March 1990.

[2] J. P. Guilford, "The Method of Pair Comparisons," in *Psychometric Methods*, McGraw-Hill, New York, 1954.

[3] R. F. Haines, "Human Performance Measurement-Validation Procedure Applicable to Advanced Manned Telescience Systems," *Research Institute for Advanced Computer Science, Technical Report No. 90.10*, February, 1990.

[4] R. F. Haines and R.W. Jackson, "Television Image Compression and Small Animal Remote Monitoring," *Research Institute for Advanced Computer Science, Technical Report No. 90.19*, April 1990.

High-Performance Compression of Astronomical Images

Richard L. White

Joint Institute for Laboratory Astrophysics, University of Colorado
Campus Box 440, Boulder, CO 80309
and
Space Telescope Science Institute
3700 San Martin Drive, Baltimore, MD 21218

rlw@stsci.edu

Summary

Astronomical images have some rather unusual characteristics that make many existing image compression techniques either ineffective or inapplicable. A typical image consists of a nearly flat background sprinkled with point sources and occasional extended sources. The images are often noisy, so that lossless compression does not work very well; furthermore, the images are usually subjected to stringent quantitative analysis, so any lossy compression method must be proven not to discard useful information, but must instead discard only the noise. Finally, the images can be extremely large. For example, the Space Telescope Science Institute has digitized photographic plates covering the entire sky, generating 1500 images each having 14000×14000 16-bit pixels. Several astronomical groups are now constructing cameras with mosaics of large CCDs (each 2048×2048 or larger); these instruments will be used in projects that generate data at a rate exceeding 100 MBytes every 5 minutes for many years.

An effective technique for image compression may be based on the H-transform (Fritze *et al.* 1977). The method that we have developed can be used for either lossless or lossy compression. The digitized sky survey images can be compressed by at least a factor of 10 with no noticeable losses in the astrometric and photometric properties of the compressed images. The method has been designed to be computationally efficient: compression or decompression of a 512×512 image requires only 4 seconds on a Sun SPARCstation 1. The algorithm uses only integer arithmetic, so it is completely reversible in its lossless mode, and it could easily be implemented in hardware for space applications.

1. Introduction

Astronomical images consist largely of empty sky. Compression of such images can reduce the volume of data that it is necessary to store (an important consideration for large scale digital sky surveys) and can shorten the time required to transmit images (useful for remote observing or remote access to data archives.)

Data compression methods can be classified as either "lossless" (meaning that the original data can be reconstructed exactly from the compressed data) or "lossy" (meaning that the uncompressed image is not exactly the same as the original.) Astronomers often insist that they can accept only lossless compression, in part because of conservatism, and in part because the familiar lossy compression methods sacrifice some information that is

needed for accurate analysis of image data. However, since all astronomical images contain noise, which is inherently incompressible, lossy compression methods produce much better compression results.

A simple example may make this clear. One of the simplest data compression techniques is run-length coding, in which runs of consecutive pixels having the same value are compressed by storing the pixel value and the repetition factor. This method is used in the standard compression scheme for facsimile transmissions. Unfortunately, it is quite ineffective for lossless compression of astronomical images because even though the sky is *nearly* constant, the noise in the sky ensures that only very short runs of equal pixels occur. The obvious way to make run-length coding more effective is to force the sky to be exactly constant by setting all pixels below a threshold (chosen to be just above the sky) to the mean sky value. However, then one has lost any information about objects close to the detection limit. One has also lost information about local variations in the sky brightness, which severely limits the accuracy of photometry and astrometry on faint objects. Worse, there may be extended, low surface brightness objects that are not detectable in a single pixel but that are easily detected when the image is smoothed over a number of pixels; such faint structures are irretrievably lost when the image is thresholded to improve compression.

2. The H-transform

Fritze *et al.* (1977; see also Richter 1978 and Capaccioli *et al.* 1988) have developed a much better compression method for astronomical images based on what they call the *H-transform* of the image. A similar transform called the S-transform has also been used for image compression (Blume & Fand 1989). The H-transform is a two-dimensional generalization of the Haar transform (Haar 1910). The H-transform is calculated for an image of size $2^N \times 2^N$ as follows:

- Divide the image up into blocks of 2×2 pixels. Call the 4 pixels in a block a_{00}, a_{10}, a_{01}, and a_{11}.
- For each block compute 4 coefficients:

$$\begin{aligned} h_0 &= (a_{11} + a_{10} + a_{01} + a_{00})/2 \\ h_x &= (a_{11} + a_{10} - a_{01} - a_{00})/2 \\ h_y &= (a_{11} - a_{10} + a_{01} - a_{00})/2 \\ h_c &= (a_{11} - a_{10} - a_{01} + a_{00})/2 \end{aligned}$$

- Construct a $2^{N-1} \times 2^{N-1}$ image from the h_0 values for each 2×2 block. Divide that image up into 2×2 blocks and repeat the above calculation. Repeat this process N times, reducing the image in size by a factor of 2 at each step, until only one h_0 value remains.

This calculation can be easily inverted to recover the original image from its transform. The transform is exactly reversible using integer arithmetic if one does not divide by 2 for

the first set of coefficients. It is straightforward to extend the definition of the transform so that it can be computed for non-square images that do not have sides that are powers of 2. The H-transform can be performed in place in memory and is very fast to compute, requiring about $16M^2/3$ (integer) additions for a $M \times M$ image.

The H-transform is a simple 2-dimensional wavelet transform. It has several advantages over some other wavelet transforms that have been applied to image compression (*e.g.*, Daubechies 1988). First, the transform can be performed entirely with integer arithmetic, making it exactly reversible. Consequently it can be used for either lossless or lossy compression (as indicated below) and one does not need a special technique for the case of lossless compression (as was required, *e.g.*, , for the JPEG compression standard.)

A second major advantage is that the H-transform is a natively 2-dimensional wavelet transform. The standard 1-dimensional wavelet transforms are extended to two dimensions by transforming the image first along the rows, then along the columns. Unfortunately, this generates many wavelet coefficients that are high frequency (hence localized) in the x-direction but low frequency (hence global) in the y-direction. Such coefficients are counter to the philosophy of the wavelet transform: high-frequency basis functions should be confined to a relatively small area of the image. Discarding these mixed-scale terms, which may be negligible compared to the noise, generates very objectionable artifacts around point sources and edges in the image. The H-transform, on the other hand, is a fully 2-dimensional wavelet transform, with all high frequency terms being completely localized. It is consequently more suitable for image compression and produces fewer artifacts.

A possible disadvantage of the H-transform is that other wavelet transforms take better advantage of the continuity of pixel values within images, so that they can produce higher compressions for very smooth images. However, for astromical images (which are mostly flat sky sprinkled with point sources) the smoothness built into higher-order transforms can actually reduce the effectiveness of compression, because one must keep more coefficients to describe each point source.

3. Compression Using the H-transform

If the image is nearly noiseless, the H-transform is somewhat easier to compress than the original image because the differences of adjacent pixels (as computed in the H-transform) tend to be smaller than the original pixel values for smooth images. Consequently fewer bits are required to store the values of the H-transform coefficients than are required for the original image. For very smooth images the pixel values may be constant over large regions, leading to transform coefficients that are zero over large areas.

Noisy images still do not compress well when transformed, though. Suppose there is noise σ in each pixel of the original image. Then from simple propagation of errors, the noise in each of the H-transform coefficients is also σ. To compress noisy images, divide each coefficient by $S\sigma$, where $S \sim 1$ is chosen according to how much loss is acceptable. This reduces the noise in the transform to $0.5/S$, so that large portions of the transform are zero (or nearly zero) and the transform is highly compressible.

Why is this better than simply thresholding the original image? As discussed above, if we simply divide the image by σ then we lose all information on objects that are within

1σ of sky in a *single* pixel, but that are detectable by averaging a *block* of pixels. On the other hand, in dividing the H-transform by σ, we preserve the information on any object that is detectable by summing a block of pixels! The quantized H-transform preserves the mean of the image for every block of pixels having a mean significantly different than that of neighboring blocks of pixels.

As an example, Figure 1 shows a 128×128 section (3.6×3.6 arcmin) from a digitized version of the Palomar Observatory–National Geographic Society Sky Survey plate containing the Coma cluster of galaxies. Figures 2, 3, and 4 show the resulting image for $S \simeq 0.5$, 1, and 2. These images are compressed by factors of 10, 20, and 60 using the coding scheme described below. In all cases a logarithmic gray scale is used to show the maximum detail in the image near the sky background level; the noise is clearly visible in Figure 1. The image compressed by a factor of 10 is hardly distinguishable from the original. In quantizing the H-transform we have adaptively filtered the original image by discarding information on some scales and keeping information on other scales. This adaptive filtering is most apparent for high compression factors (Fig. 4), where the sky has been smoothed over large areas while the images of stars have hardly been affected.

The adaptive filtering is, in itself, of considerable interest as an analytical tool for images (Capaccioli *et al.* 1988). For example, one can use the adaptive smoothing of the H-transform to smooth the sky without affecting objects detected above the (locally determined) sky; then an accurate sky value can be determined by reference to any nearby pixel.

The blockiness that is visible in Figure 4 is the result of difference coefficients being set to zero over large areas, so that blocks of pixels are replaced by their averages. It is possible to eliminate the blocks by an appropriate filtering of the image. A simple but effective filter can be derived by simply adjusting the H-transform coefficients as the transform is inverted to produce a smooth image; as long as changes in the coefficients are limited to $\pm S\sigma/2$, the resulting image will still be consistent with the thresholded H-transform.

4. Efficient Coding

The quantized H-transform has a rather peculiar structure. Not only are large areas of the transform image zero, but the non-zero values are strongly concentrated in the lower-order coefficients. The best approach we have found to code the coefficient values efficiently is quadtree coding of each bitplane of the transform array. Quadtree coding has been used for many purposes (see Samet 1984 for a review); the particular form we are using was suggested by Huang and Bijaoui (1991) for image compression.

- Divide the bitplane up into 4 quadrants. For each quadrant code a '1' if there are any 1-bits in the quadrant, else code a '0'.
- Subdivide each quadrant that is not all zero into 4 more pieces and code them similarly. Continue until one is down to the level of individual pixels.

This coding (which Huang and Bijauoi call "hierarchic 4-bit one" coding) is obviously very well suited to the H-transform image because successively lower orders of the H-transform coefficients are located in successively divided quadrants of the image.

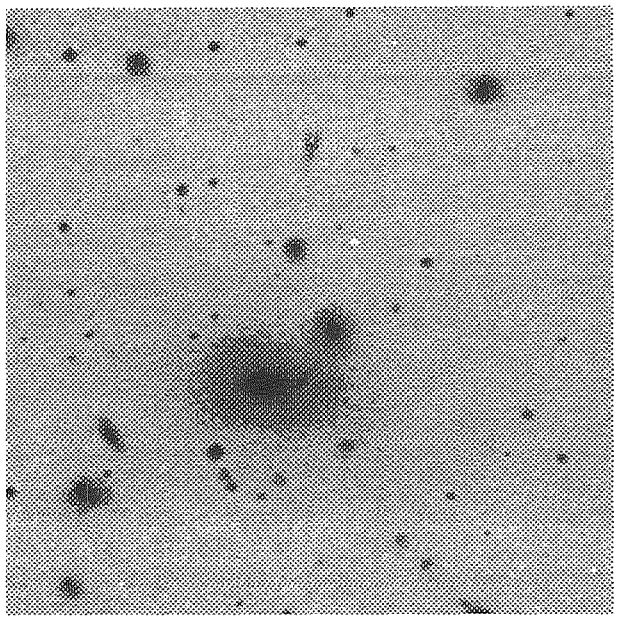


Figure 1. Original image of Coma cluster.

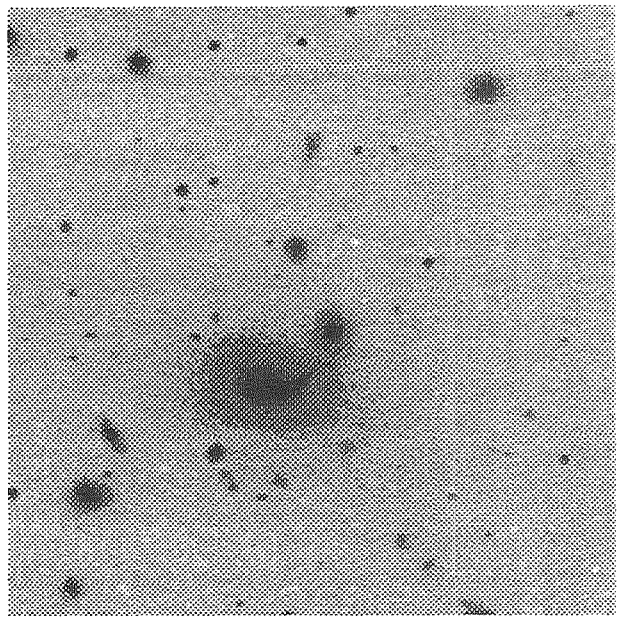


Figure 2. Result of compression by factor of 10.

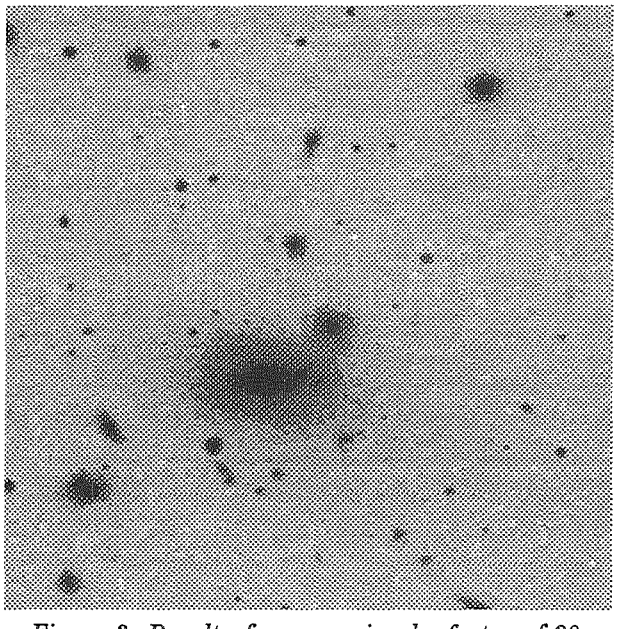


Figure 3. Result of compression by factor of 20.

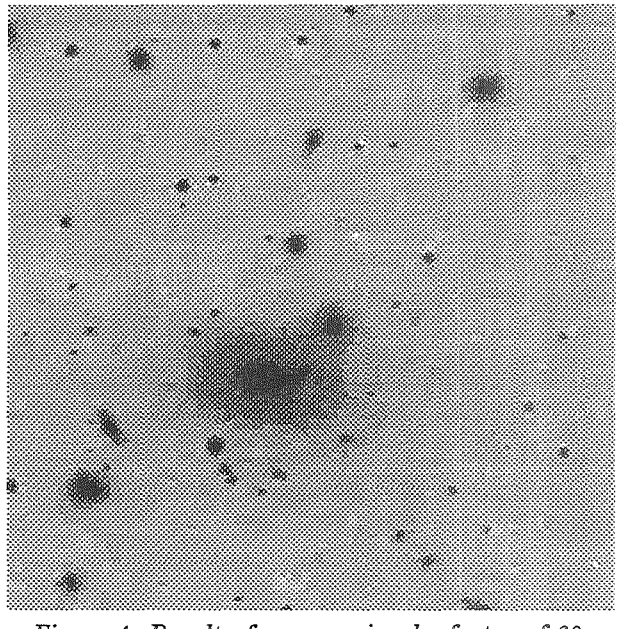


Figure 4. Result of compression by factor of 60.

We follow the quadtree coding with a fixed Huffman coding that uses 3 bits for quadtree values that are common (*e.g.*, 0001, 0010, 0100, and 1000) and uses 4 or 5 bits for less common values. This reduces the final compressed file size by about 10% at little computational cost. Slightly better compression can be achieved by following quadtree coding with arithmetic coding (Witten, Bell, and Cleary 1987), but the CPU costs of arithmetic coding are not, in our view, justified for 3–4% better compression. We have also tried using arithmetic coding directly on the H-transform, with various contexts of neighboring pixels, but find it to be both computationally inefficient and not significantly better than quadtree coding.

For completely random bitplanes, quadtree coding can actually use more storage than simply writing the bitplane directly; in that case we just dump the bitplane with no coding.

Note that by coding the transform one bitplane at a time, the compressed data can be viewed as an incremental description of the image. One can initially transmit a crude representation of the image using only the small amount of data that is required for the sparsely populated, most significant bit planes. Then the lower bit planes can be added one by one until the desired accuracy is required. This could be useful, for example, if the data is to be retrieved from a remote database — one could examine the crude version of the image (retrieved very quickly) and abort the transmission of the rest of the data if the image is judged to be uninteresting.

5. Astrometric and Photometric Properties of Compressed Images

Astronomical images are not simply subjected to visual examination, but are also subjected to careful quantitative analysis. For example, for the image in Figure 1 one would typically like to do astrometric (positional) measurements of objects to an accuracy much better than 1 pixel, photometric (brightness) measurements of objects to an accuracy limited only by the detector response and the noise, and accurate measurements of the surface brightness of extended sources.

We have done some experiments to study the degradation of astrometry and photometry on the compressed images compared to the original images (White, Postman, and Lattanzi 1991). Even the most highly compressed images have very good photometric properties for both point sources and extended sources; indeed, photometry of extended objects can be improved by the adaptive filtering of the H-transform (Cappacioli *et al.* 1988). Astrometry is hardly affected by the compression for modest compression factors (up to about a factor of 20 for our digitized photographic plates), but does begin to degrade for the most highly compressed images.

These results are based on tests carried out with tools optimized for the original images; it is likely the best results will be obtained for highly compressed images only with analysis tools specifically adapted to the peculiar noise characteristics of the compressed images.

6. Conclusions

In order to construct the Guide Star Catalog for use in pointing the Hubble Space Telescope, the Space Telescope Science Institute scanned and digitized wide-field photographic plates covering the entire sky. The digitized plates are of great utility, but to date it has been impossible to distribute the scans because of the massive volume of data involved (a total of about 600 Gbytes). Using the compression techniques described in this paper, we plan to distribute our digital sky survey on CD-ROMs; about 100 CD-ROMs will be required if the survey is compressed by a factor of 10.

The algorithm described in this paper has been shown to be capable of producing highly compressed images that are very faithful to the original. Algorithms designed to work on the original images can give comparable results on object detection, astrometry,

and photometry when applied to the images compressed by a factor of 10 or possibly more. Further experiments will determine more precisely just what errors are introduced in the compressed data; it is possible that certain kinds of analysis will give more accurate results on the compressed data than on the original because of the adaptive filtering of the H-transform (Capaccioli *et al.* 1988).

This compression algorithm can be applied to any image, not just to digitized photographic plates. Experiments on CCD images indicate that lossless compression factors of 3–30 can be achieved depending on the CCD characteristics (*e.g.*, the readout noise). A slightly modified algorithm customized to the noise characteristics of the CCD will do better. This application will be explored in detail in the future.

We gratefully acknowledge grant from NAGW-2166 from the Science Operations Branch of NASA headquarters which supported this work. The Space Telescope Science Institute is operated by AURA with funding from NASA and ESA.

References

Blume, H., and Fand, A. 1989, *SPIE Vol. 1091, Medical Imaging III: Image Capture and Display*, p. 2.

Capaccioli, M., Held, E. V., Lorenz, H., Richter, G. M., and Ziener, R. 1988, *Astronomische Nachrichten*, **309**, 69.

Daubechies, I. 1988, *Comm. Pure and Appl. Math.*, **41**, 909.

Fritze, K., Lange, M., Möstl, G., Oleak, H., and Richter, G. M. 1977, *Astronomische Nachrichten*, **298**, 189.

Haar, A. 1910, *Math. Ann.* **69**, 331.

Huang, L, and Bijaoui, A. 1991, *Experimental Astronomy*, **1**, 311.

Richter, G. M. 1978, *Astronomische Nachrichten*, **299**, 283.

Samet, H. 1984, *ACM Computing Surveys*, **16**, 187.

White, R. L., Postman, M., and Lattanzi, M. 1991, in *Proceedings of the Edinburgh Meeting on Digital Sky Surveys*, in press.

Witten, I. H., Radford, M. N., and Cleary, J. G. 1987, *Communications of the ACM*, **30**, 520.

MULTI-RATE, REAL TIME IMAGE COMPRESSION FOR IMAGES DOMINATED BY POINT SOURCES

A. Kris Huber, Scott E. Budge, and Richard W. Harris
Department of Electrical Engineering
Utah State University
Logan, Utah 84322-4120

Abstract. An image compression system recently developed for compression of digital images dominated by point sources is presented. Encoding consists of minimum-mean removal, vector quantization, adaptive threshold truncation, and modified Huffman encoding. Simulations are presented showing that the peaks corresponding to point sources can be transmitted losslessly for low signal-to-noise ratios (SNR) and high point source densities while maintaining a reduced output bit rate. Encoding and decoding hardware has been built and tested which processes 552,960 12-bit pixels per second at compression rates of 10:1 and 4:1. Simulation results are presented for the 10:1 case only.

1. Introduction

Infrared images of space can be characterized by point sources such as stars and galaxies, spatially structured sources from galactic matter, moving objects such as planets and asteroids, and the background due to the zodiacal dust in the solar system [1]. We present a data compression algorithm tailored to this type of image when point sources and/or small moving objects are the primary features of interest.

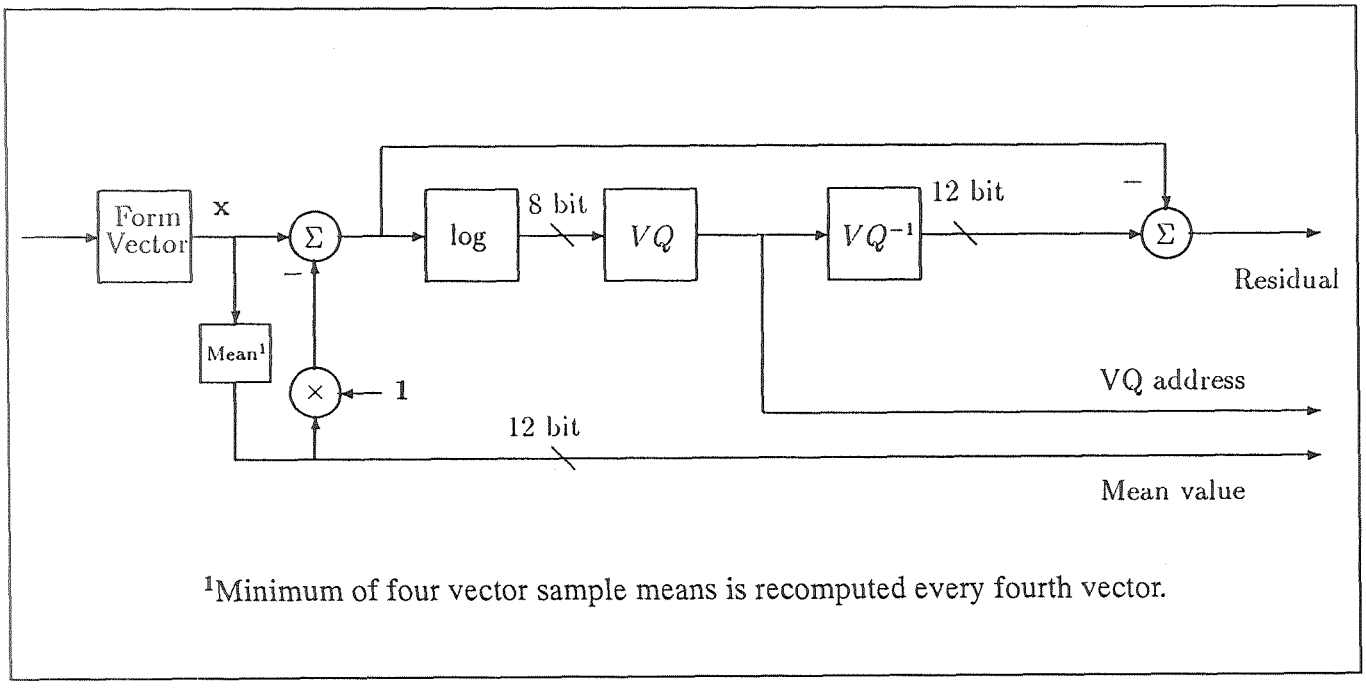
The compression algorithm involves a combination of a the lossy algorithm, mean-residual vector quantization (MRVQ) [2,3,4], and the lossless algorithms run-length encoding and Huffman encoding [5,6]. While moderate-sized MRVQ encodes many scenes very well, including space backgrounds of low spatial frequency, MRVQ is not efficient for encoding point sources, since they are characterized by very high spatial frequencies. Thus the intent of the lossless stage is to encode the largest of the MRVQ errors and transmit as many of them as possible from the space platform using the telemetry bandwidth not needed by the MRVQ.

Section 2 discusses the compression algorithm. Section 3 describes the scene and instrument models used for the experiments of Section 4. Section 5 is a summary and conclusion.

2. A Multi-Rate Hybrid Algorithm

The descriptions of the algorithm given in this section includes some implementation details specific to the system built at the Space Dynamics Laboratory (SD) in Logan, Utah, for the Spirit 3 infrared radiometer. Some hardware trade-off issues for this algorithm are discussed in [7]. While hardware details are distinct from the algorithm itself, they may help the reader to visualize the encoding process. Refer to Figures 1 and 2 for a block diagram of the following encoding algorithm steps:

1) **Minimum-Mean Removal**. The mean background level is estimated and subtracted from the original image. The estimate is made by computing the sample means of four 16-pixel blocks (vectors) and choosing the minimum. The minimum mean is subtracted from all four vectors, resulting in the *mean residual.* The mean value is passed to the data multiplexor (Step 7) via a first-in first-out (FIFO) buffer, along with a small amount of side information which



[1]Minimum of four vector sample means is recomputed every fourth vector.

Figure 1. Block diagram of lossy encoding section, steps 1 through 3.

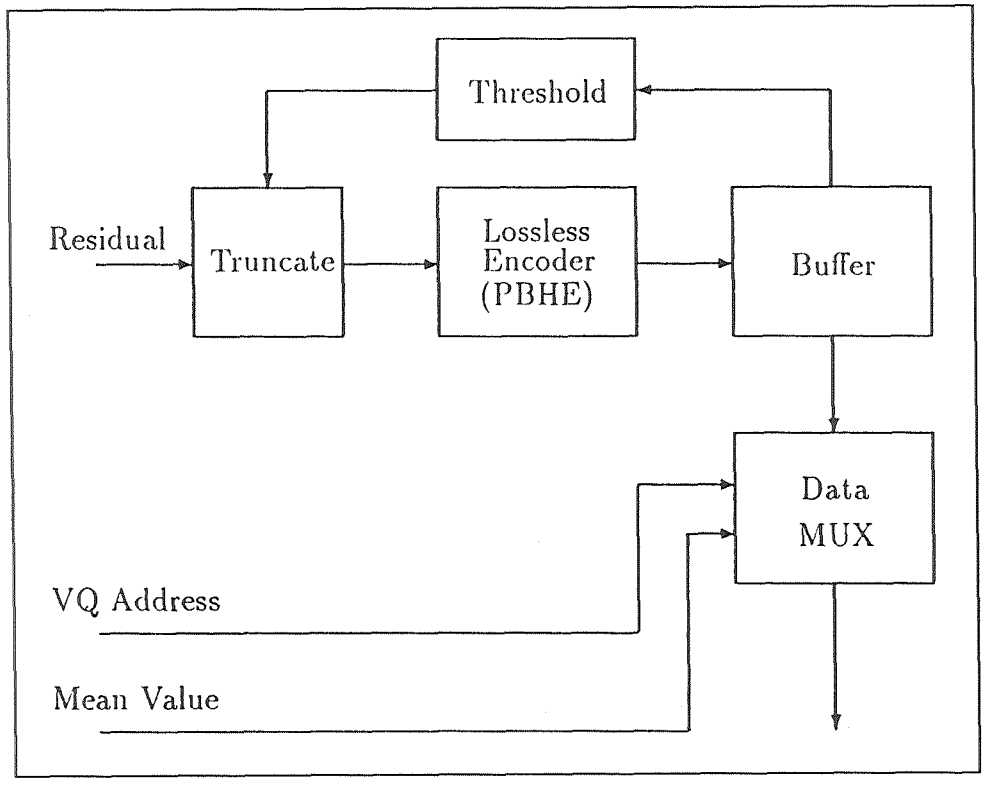


Figure 2. Block diagram of lossless stage, steps 4 through 7.

includes some key housekeeping words. From the side information the decoder determines whether the data is being compressed at a ratio of 4:1 or 10:1 and adjusts itself accordingly.

2) **Vector Quantization (VQ).** Each vector of *mean residual* is quantized to one of 256 codebook vectors. The codebook is structured as a 2-level tree with 16 vectors on the top level and 256 vectors on the bottom level. The codebook is searched real-time using a custom VLSI chip developed at Utah State University (USU) for VQ applications. The 13-bit mean residual pixels are converted to 9-bit values prior to VQ to accommodate the codebook processor chip. This is done by scaling the magnitude logarithmically using a look-up table. The magnitude of all values in the encoding codebook are also scaled in the same way (off-line). The address of the codebook vector with the least absolute difference from the residual vector is given to the data multiplexor (Step 7) via a FIFO buffer.

3) **Residual Formation.** The *residual* is the difference between the input vector and the MRVQ result. It is formed by an inverse VQ process and subtraction from the *mean residual*. The inverse VQ process involves accessing the decoding codebook memory to look up the 12-bit (magnitude not scaled) vector corresponding to the codebook vector chosen in Step 2.

4) **Adaptive Threshold Truncation.** The *residual* is compared pixel-by-pixel against a threshold which depends on the instantaneous fullness level of the buffer of lossless data (see Step 5). If the threshold is exceeded, the residual value is sent to the next encoding step, otherwise 0 is sent to the next step. The image formed by these values is called the *truncated residual.*

5) **Pixel-Based Huffman Encoding (PSYCHE).** The *truncated residual* is encoded losslessly by a combination of run-length and constrained Huffman coding. The codewords are precomputed off-line using an algorithm for constructing codewords of limited maximum length [6]. A special "header" codeword is reserved to prefix very large or very unlikely residual values. This reduces the amount of memory required to store the codewords. Encoding involves grouping the zeroes into runs and retrieving variable-length codewords from memory to represent runs and residual values. Codewords are packed end-to-end and placed in a FIFO buffer. The buffer is also accessed by the data multiplexor (Step 7), which empties it at a constant rate that depends on the selected compression ratio. The buffer fullness level is used to establish the adaptive threshold of Step 4.

6) **Synchronization and Error Detection.** At the end of each frame a synchronization codeword is transmitted, followed by a frame count. The synchronization codeword is such that it cannot be confused with any possible concatenations of codewords. The frame count is used by the decoder in the event of a loss of synchronization to correctly associate the lossless data frame with the MRVQ data frame. A 16-bit cyclic redundancy check (CRC) is inserted by the data multiplexor (Step 7). If the CRC computed by the decoder does not match, the lossless frame is not decoded. This feature prevents the large errors that would occur if a mis-aligned truncated residual were summed with the decoded MRVQ image.

7) **Data Multiplexing.** A predefined telemetry frame format is understood by both encoder and decoder. This format specifies the number of bits of mean, VQ, and lossless data sent in each telemetry frame. Mean values, VQ addresses and lossless data are drawn from their respective FIFO buffers according to the telemetry frame format and transmitted to Earth. The overall compression ratio is determined by the number of bits per frame into the compressor divided by the number of bits in the telemetry frame. Therefore adjustments can be made to the compression ratio by adjusting either the input data rate and/or the telemetry frame itself. Two modes are defined in the Spirit 3 compressor with compression ratios of 10:1 and 4:1. The difference between the two modes is that the 4:1 mode has a slower input rate and a telemetry frame with a higher proportion of lossless data. This multiple-rate capability is necessary or

desirable in many image compression systems, particularly for sensors which have the capability of sampling at various rates but transmit their data through a telemetry channel of fixed bandwidth.

8) **Establishment of Error Bounds.** Post processing can optionally be done to generate intervals that bound the true input signal. Whenever a non-zero residual value was transmitted losslessly the error is zero, thus the algorithm provides a subsampled image that is transmitted losslessly. Pixels where the residual was truncated by the adaptive threshold are in error by at most plus or minus the value of the threshold. Since the lossless buffer level can be determined by the delay between the MRVQ data and corresponding lossless data, the threshold can also be determined after encoding without sending side information if the threshold dependence on buffer level is known. In order to do this to full advantage, the hardware should be designed so that the signals which fill and empty the lossless FIFO are derived from the same clock, so they can be simulated exactly.

The decoding algorithm is simply the inverse of the encoding algorithm and will not be elaborated upon here.

3. Scene and Sensor Model

The hardware compression system is part of the Spirit 3 infrared radiometer built by the Space Dynamics Laboratory (SD) in Logan, Utah. This radiometer acquires data in five spectral bands from 2 to 26 microns in wavelength. An image is scanned across long rectangular focal plane arrays (FPAs) to form the scene. Some of the instrument parameters and performance estimates used in modelling the radiometer are given in Table 1. The data compressor provides quick-look capability to the instrument but is not the critical path for the data.

Table 1. Parameters used for simulating the SPIRIT 3 radiometer

		SPECTRAL BAND				
NAME	Units	A	B	C	D	E
Center Wavelength	μm	8.5	4.4	12	15	23
Bandwidth	μm	5	0.2	2.1	2.5	9.9
Integration Time	ms	2.8	2.8	2.8	2.8	2.8
Detector Field of View	sr	8.1E-09	8.1E-09	8.1E-09	8.1E-09	8.1E-09
Noise-Equivalent Radiance	W/cm^2-sr	3.1E-10	1.8E-09	2.2E-10	1.7E-10	1.8E-10

One of the missions of the instrument is to resolve natural and artificial point sources. To evaluate the performance of the algorithm under worst case conditions for point source scenes we used a scene generated by a state-of-the-art infrared scene generator, the Strategic Scene Generator Model (SSGM) version 2.3. In particular, we generated a scene of the galactic center where the point source density is the greatest. SSGM 2.3 documentation states that space scenes are subject to undercounting in the galactic plane, so the scene we generated is somewhere between the typical scene and the worst case.

In addition to the starfield and background, the sensor characteristics provide structure to the scene. In this system the data is delivered to the compressor uncalibrated, i.e. the measurements have not been corrected for the non-uniformity of response of the individual detectors in the array. Due to hardware constraints the image blocks are formed as 16x1 vectors with each pixel within a vector receiving data from the same spectral band, but from a different detector. Such non-

uniformity translates into higher entropy for the data stream and increases the importance of accurate sensor simulation when attempting to evaluate algorithm performance.

In our simulations we model the detector input, x, as

$$x = L * R_r + p_i * N * (L*R_r)^{0.5}$$

where L is the radiant flux value generated by the scene generator model (in $Wcm^{-2}sr^{-1}$), R_r is the nominal radiance responsivity (with units of counts per $Wcm^{-2}sr^{-1}$) that converts radiance to counts, N is a zero-mean unity-variance Gaussian random variable, and p_i is a detector coefficient to model the photon noise. Photon noise is a signal-dependent noise source that increases in direct proportion to the square root of the flux. Photon noise is caused by fluctuations in arrival rate of incident photons and ultimately limits the SNR [8].

We model the detector output as

$$f(x) = a_i + b_i x + c_i x^2 + N * d_i$$

where a_i, b_i and c_i are coefficients for offset, gain, and non-linearity that approximate the responsivity curve for each detector, and d_i is the standard deviation of the dark noise for each detector. The detector outputs are then uniformly quantized to obtain the 12-bit integer that is the input to the data compressor.

4. Experimental Results

Several experiments were performed to evaluate the performance of the algorithm. As noted in [9], average RMS error is statistically precise, but not always appropriate. In particular, large localized errors can occur with lossy compression techniques which do not increase the average error very much, but drastically diminish the quality of the compressed image by other standards. Therefore, in these simulations the distortion (in terms of RMS error) was tracked for pixels corresponding to point source peaks and reported separately from the distortion to pixels corresponding to background. All simulations were of the mode in which the compression ratio is 10:1 (1.2 bits per pixel).

Simulation #1 included preliminary estimates of focal plane coefficients. A 1 deg^2 infrared background scene for Simulation #1 was generated internally before the SSGM became available at SD.

For Simulations #2 and #3 the focal plane coefficients were based upon noise and responsivity tests performed by the focal plane vendors. Compared to the estimates of Simulation #1 the offset coefficients were less uniform, but the gain coefficients were more uniform. The infrared background scene generated by the SSGM was used for these simulations.

4.1 Simulation #1.

In this simulation the sensor coefficients were based on preliminary estimated values with the a_i, b_i, and c_i uniformly distributed over the intervals [10, 20], [0.8, 1.2], and $[-10^{-6}, 0]$, respectively. The dark noise, d_i, was modelled as uniform and equal to 1. The photon noise was also uniform across the detectors.

The detector non-uniformities effectively increased the noise sigma at the input to the data compressor from 1.0 to 5.5 counts. For this reason the point source SNR was referenced to the latter noise level. Point sources were positioned randomly and uniformly throughout the image. Their amplitude at the detector output divided by the noise sigma was considered to be the peak SNR of the point source.

Pre-processing consisted of generating an MRVQ codebook and PSYCHE codewords, which were shared by encoder and decoder. The residual codebook was generated using the LBG algorithm [4] from a training set consisting of the mean residual of a different background image. The PSYCHE codewords were based on codeword probabilities which were the average of probabilities estimated during preliminary runs of a different image with 12,000 and 24,000 point source pixels and SNR ranging from 5 to 30. The codeword lengths were computed using a modified form of the Huffman algorithm of Murakami, et. al. [6].

Root-mean-square (RMS) error in background pixels was nearly constant over the entire ensemble of simulation images as shown in Figure 3. Error in the point-source pixels is shown in Figure 4. No error occurred to point sources for SNR greater than 7 and number less than 24,000. Since the Spirit 3 radiometer scans a scene with four samples per dwell, and each point source is seen by all 8 FPA columns that are compressed, 24,000 point source pixels corresponds to a point source density of 750 per deg^2. For SNR less than 7, error in point source pixels and background pixels was comparable. As the number of point sources increased beyond 750 per deg^2, a substantial increase in error occurred which became higher with SNR (Figure 4). At such high densities PSYCHE was unable to encode all of the point source pixels with the allocated bandwidth. Those which were transmitted were still error-free.

4.2 Simulation #2.

In this simulation the sensor coefficients were based on roughly measured values (the data used to characterize the focal plane was not intended for calibration), but were probably closer to reality than in Simulation #1. The estimated means and standard deviations for the nominal detectors of the five focal plane arrays corresponding to the 5 spectral bands are given in Table 2. Where data was not available (indicated by n/a) a uniform focal plane was assumed (standard deviation of 0) and an average of available data was used for the mean.

Table 2. Sensor Parameters for Simulation #2

		SPECTRAL BAND					
		A	B	C	D	E	UNITS
a_i	Mean:	161	135	107	84	84	cnt
	Std:	39	n/a	19	23	17	cnt
b_i	Mean:	1.0	1.0	1.0	1.0	1.0	cnt/cnt
	Std:	n/a	n/a	.015	.020	.017	
c_i	Mean:	-10^{-5}	-10^{-5}	-10^{-5}	-10^{-5}	-10^{-5}	cnt/sqrt(cnt)
	Std:	n/a	n/a	n/a	n/a	n/a	
d_i	Mean:	0.6	0.6	0.9	0.9	0.9	cnt rms
	Std:	n/a	n/a	0.17	0.22	0.21	
p_i	Mean:	n/a	n/a	.128	.123	.122	cnt/sqrt(cnt)
	Std:	n/a	n/a	.009	.008	.008	

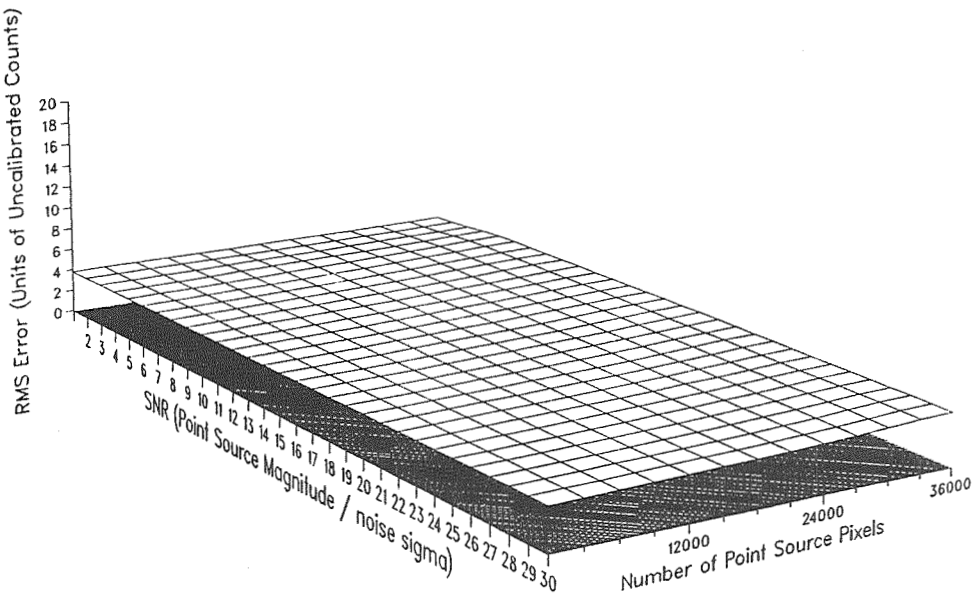


Figure 3. Background error for varying SNR and number of sources, Simulation #1.

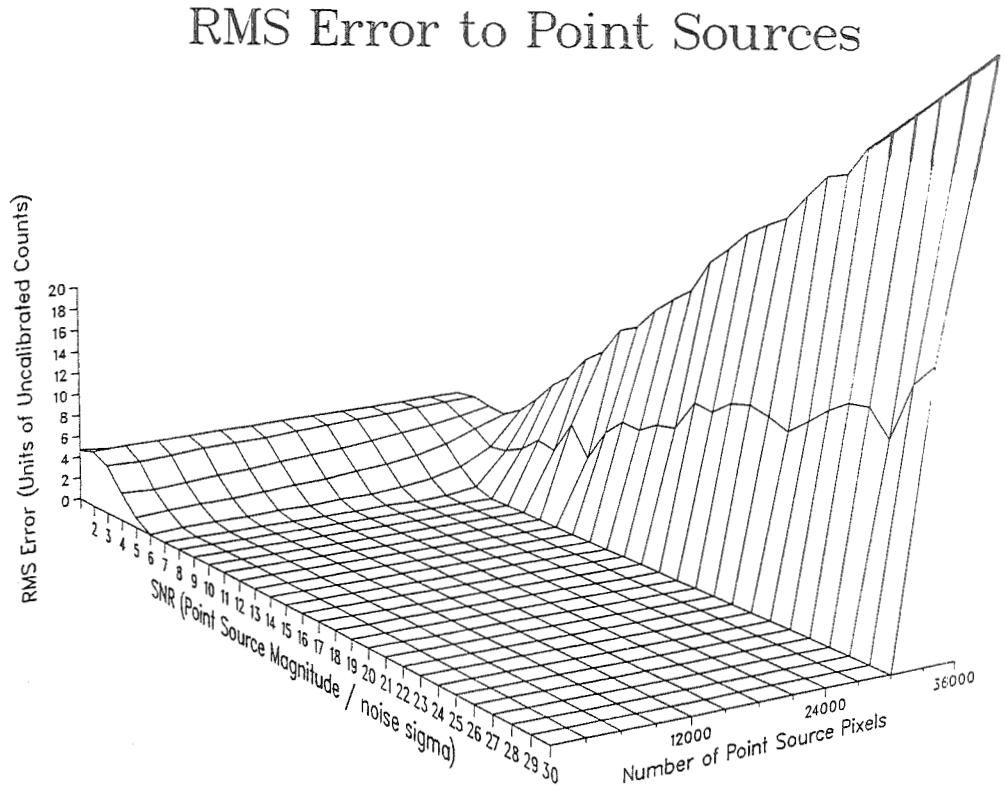


Figure 4. Point source error for varying SNR and number of sources, Simulation #1. 24,000 point source pixels corresponds to 750 sources per deg^2.

Separate images from the SSGM were used to train the compressor from those used to obtain performance results. As in Simulation #1, additional point sources of constant SNR were positioned randomly to evaluate how algorithm performance varied with point source density. Figure 5 shows the RMS error due to compression as a function of point source SNR and number density. These indicate that most of the point source radiance peaks are encoded losslessly for SNR above about 60 and point source density below about 300 per deg^2. We believe the reason the average error in this region is non-zero is due to the few bad detectors in the focal plane arrays. The compression error to the background remained nearly constant at 4.6 counts until point source density became very high.

Note that in this simulation and the next, the definition of SNR was changed. The SNR was defined as the peak point source radiance in counts divided by the RMS dark noise, d_i. In other words we used the detector SNR and did not include the pseudo-noise at the compressor input due to non-uniformity as we did in Simulation #1.

Figure 6 shows the result when the simulation was repeated for a more ideal case. The dark noise was made equal for all spectral bands to 1 count rms, and the focal planes were assumed to be perfectly uniform (coefficients were set to the mean value for the nominal detectors). A significant improvement resulted. The point source peak radiance was encoded losslessly for SNR above about 8 and point source density below about 1000 per deg^2. The background was encoded with RMS error of about 0.9 counts in this region.

4.3 Simulation #3.

In this simulation the sensor coefficients were the same as in Simulation #2. The original SSGM scene of the galactic center was compressed with and without non-uniformities. As an add-hoc method of compensating for the point source undercounting in our version of SSGM, we doubled the radiance of the SSGM scene. This gave a point source density of about 188 per deg^2 for sources with an SNR of 3 and greater.

Figure 7 shows the RMS error to point sources when FPA non-uniformity was included. In this case point source peaks were transmitted losslessly for sources of SNR greater than 36. Background error was 4.5 counts rms.

Figure 8 shows the RMS error to point sources for the same scene imaged with uniform detector arrays and dark noise of 1 count rms. This caused the threshold above which point source peaks were transmitted losslessly to be reduced to 7. Background error in this case was reduced to 0.84 counts rms.

5. Summary and Conclusion

A hybrid lossy/lossless algorithm for compression of space images dominated by point sources was described. State of the art simulation software was used to model the infrared sky for the spectral bands of the radiometer. The model used to simulate some of the dominant sensor characteristics was described. The data derived from these models fed several data compression experiments in which the images were segmented into pixels corresponding to point sources and pixels corresponding to background. Errors caused by the compression algorithm operating at a compression ration of 10:1 were tracked separately for point sources and background.

The simulations showed that background was encoded quite well, but that focal plane uniformity could significantly reduce the errors due to the compression algorithm. A conclusion that can be drawn from this is that it is desirable to compress calibrated data when this is feasible.

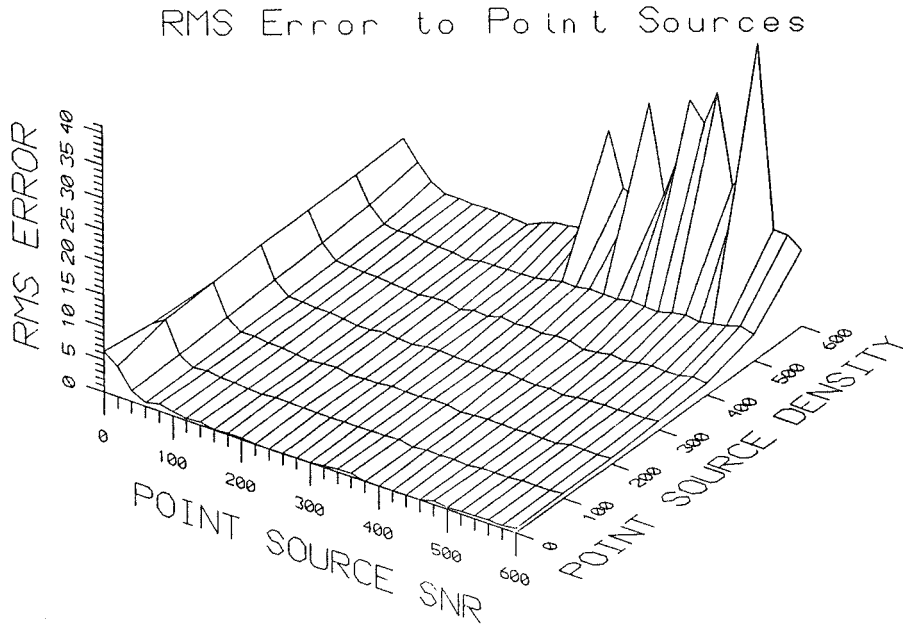


Figure 5. Simulation #2. RMS error (in counts) to point source peak values due to 10:1 compression. SNR was defined as the point source peak magnitude divided by the focal plane array dark noise coefficients. Non-uniformity of FPAs was simulated based on measured data. Background error was nearly constant at about 4.6 counts.

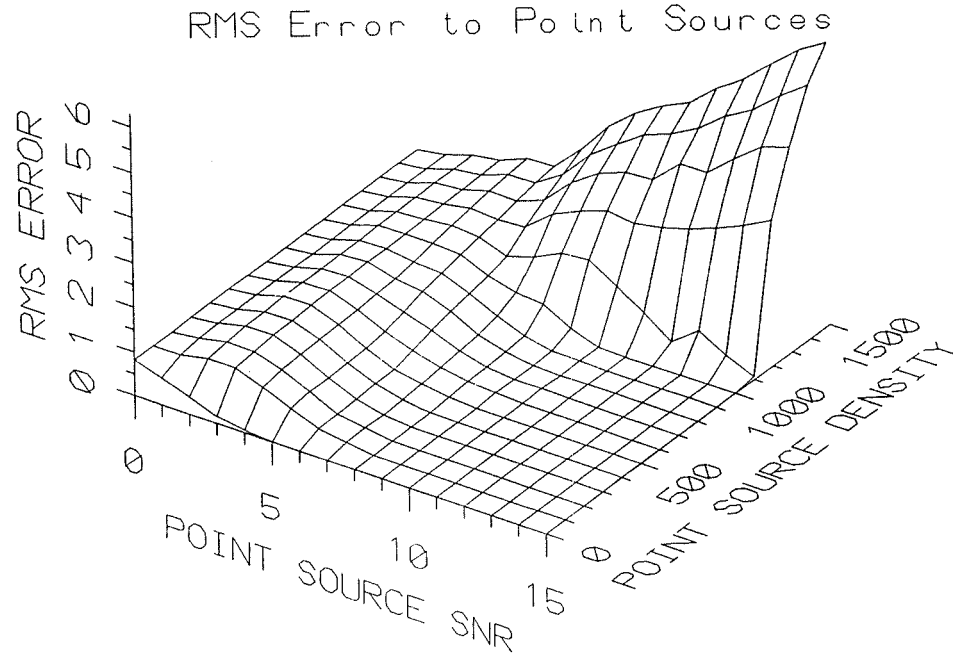


Figure 6. Simulation #2. RMS error (in counts) to point source peak values due to 10:1 compression. SNR was defined as the point source peak magnitude divided by the focal plane array dark noise. The mean offset and gain coefficients were used for all detectors of an array, thus simulating perfect uniformity. The background error was nearly constant at about 0.9 counts for the region plotted.

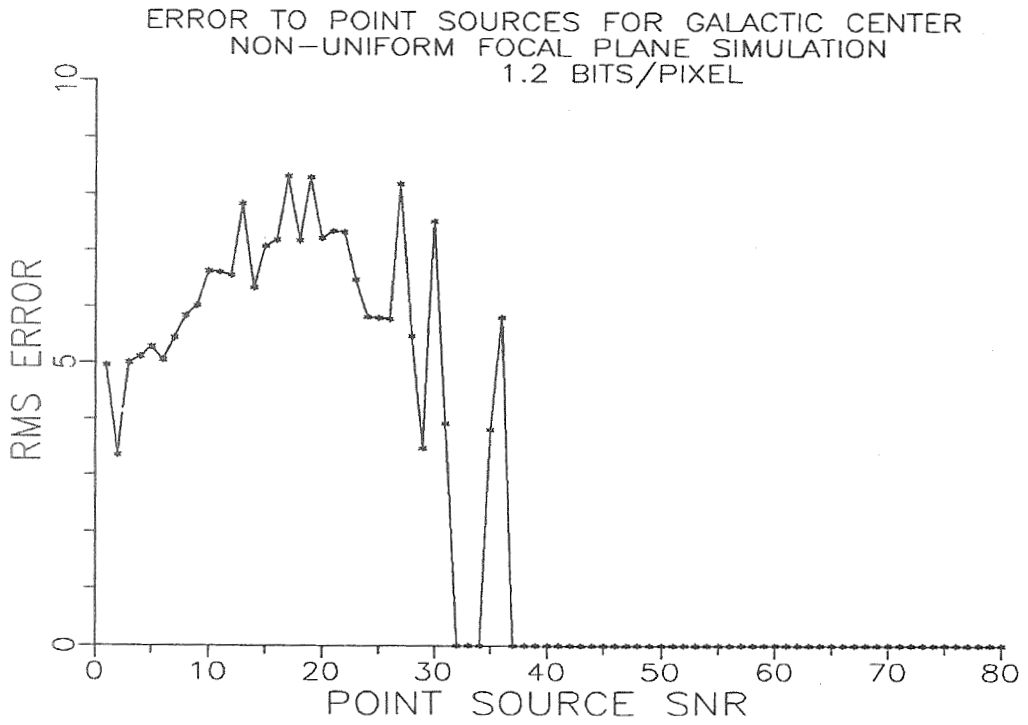


Figure 7. Simulation #3. RMS error to point sources in the galactic scene imaged onto non-uniform focal plane arrays. Background error was 4.5 counts rms.

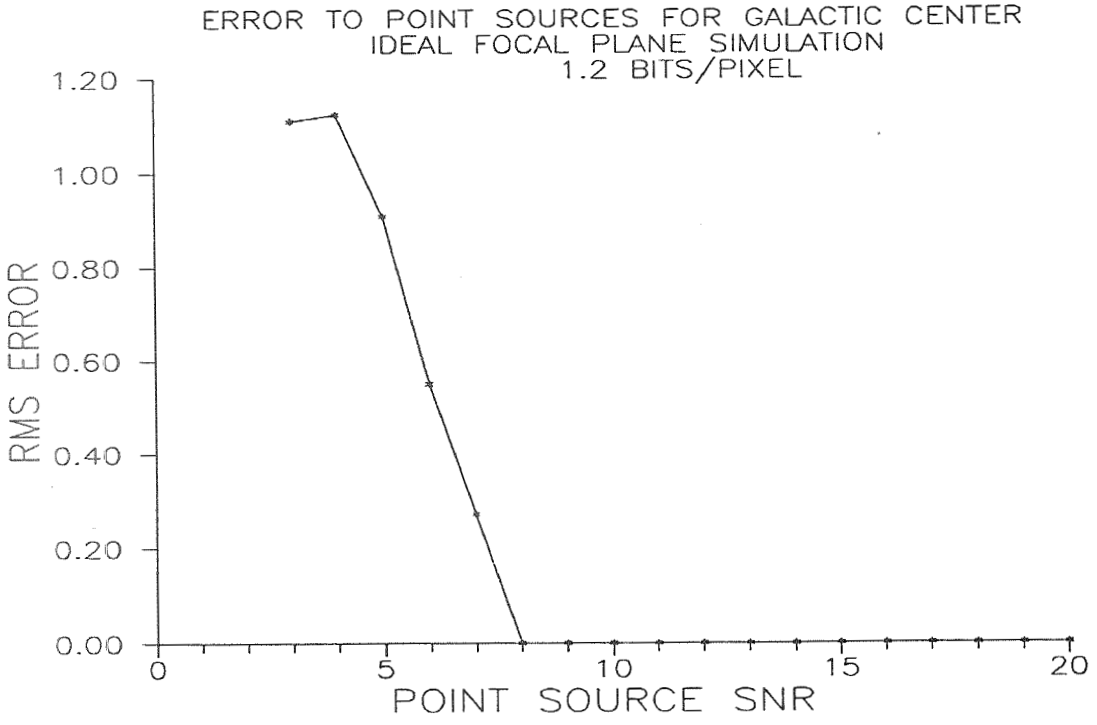


Figure 8. Simulation #3. RMS error to point sources in the galactic scene imaged onto perfectly uniform focal plane arrays. Background error was 0.84 counts rms.

The simulations also showed that no error occurred to peaks of point sources over a wide range of point source densities and relatively low SNR. The trends of the graphs generated in Simulations #1 and #2 held in Simulation #3, namely low average error to background and lossless transmission of point source peaks detected with sufficient SNR. Excellent performance in that simulation indicates that the compressor should perform well when the sensor is sampling scenes that are dominated by point sources, such as those simulated. Reduction of simulated performance is expected due to optical blurring which was not simulated. Blurring will reduce the point source peak SNR and effectively increase the point source density.

Further research is needed to investigate the bounded error property of this algorithm and design enhancements and new algorithms that use this property to increase coding efficiency. Additionally, scientific algorithms for data reduction that use the time-varying compression error bound could be designed.

Acknowledgment

This work was fully funded by the Space Dynamics Laboratory in Logan, Utah. We wish to thank Clair Wyatt (SD), Wayne Nichols (SD), and Howard Cole (NRC) for assistance in obtaining the Spirit 3 instrument data and simulated infrared scenes.

References

[1] Price, S. D. and Kenneally, J. P., "The Infrared Celestial Background Scene Descriptor," *Imaging Infrared: Scene Simulation, Modeling, and Real Image Tracking*. A. J. Huber, M. Triplett, J. Wolverton, Editors, Proc. SPIE 1110, pp. 38-48, 1989.

[2] Baker, R. L., "Vector Quantization of Digital Images," Ph.D. Dissertation, Stanford Univ., 1984.

[3] Gray, R. M., "Vector Quantization," *IEEE ASSP Magazine*, pp. 4-29, Apr. 1984.

[4] Linde, Y., Buzo, A., and Gray, R. M., "An Algorithm for Vector Quantizer Design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84-95, Jan. 1980.

[5] Huffman, D. A., "A Method for the Construction of Minimum-Redundancy Codes," *Proc. IRE*, vol. 40, pp. 1098-1101, Sept. 1952.

[6] Murakami, H., Matsumoto, S., and Yamamoto, H., "Algorithm for Construction of Variable Length Code with Limited Maximum Word Length," *IEEE Trans. Commun.*, vol. COM-32, pp. 1157-1159, Oct. 1984.

[7] Singer, J. H., "System Implementation Trade-Offs of Lossless Data Compression Algorithms for Space Sensor Image Compression," M.S. Thesis, Utah State Univ., 1991.

[8] Wyatt, C. L., *Electro-Optical System Design for Information Processing*, McGraw-Hill: New York, 1991.

[9] Erickson, D. E., "Data Compression for Near Earth and Deep Space to Earth Transmission," *Space and Earth Science Data Compression Workshop*. J. C. Tilton, Editor, NASA Conference Publication 3130, Nov. 1991.

APPENDIX

WORKSHOP PARTICIPANTS

Space and Earth Science Data Compression Workshop
Snowbird, Utah - March 27, 1992

Held in conjunction with the Data Compression Conference (DCC'92)

Maaruf Ali
Dept. of Electrical & Electrical Eng.
King's College London
WC2R 2LS
UNITED KINGDOM
011-44-71-836-5454 ext. 3583
m.ali@uk.ac.kcl.cc.oak

John Baras
Systems Research Center
University of Maryland
A.V. Williams Buildling
College Park, MD 20742
301-405-6606
baras@src.umd.edu

Tom Bizon
NASA/Lewis Research Center
21000 Brookpark Road
M/S 54-8
Cleveland, OH 44135
216-433-8121
cabizon@lims05.lerc.nasa.gov

Anselm Blumer
Computer Science Dept.
Tufts University
Medford, MA 02155
617-627-3651
ablumer@cs.tufts.edu

Ronald Boyd
CTA Inc.
6116 Executive Blvd.
Suite 800
Rockville, MD 20852
301-816-1289
boyd@cta.com

Jonathan Bradley
505 Oppenheimer Drive
Los Alamos, NM 87544
512-662-3687
bradley@lanl.gov

Joe Bredekamp
NASA Headquarters
Code SM
Washington, DC 20546
202-453-1505
202-755-9234 (Fax)
jbredekamp@nasamail.nasa.gov

Scott Budge
Electrical Eng. Dept.
Utah State University
Logan, UT 84322-4120
801-750-3433
scott@goga.ee.usu.edu

Thomas Caverly
Computer Science Dept.
Rancho Santiago College
17th & Bristol
Santa Ana, CA 92706
714-564-6766

Kar-Ming Cheung
Mail Stop 238-420
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109
818-393-9480
kmcheung@galois.jpl.nasa.gov

Anatoli Chklovski
Digital Biometrics
5600 Rowland Road
Suite 205
Minnetonka, MN 55343
612-932-0888

Sherry Chuang
NASA/Ames Research Center
MS 269-4
Moffett Field, CA 94035-1000
415-604-3376
chuang@ptolemy.arc.nasa.gov

Tom Cover
Stanford University
Durand Building
Room 121
Stanford, CA 94305
415-723-4505
cover@isl.stanford.edu

David Craft
Internal Zip 9560
IBM Austin Laboratory
11400 Burnet Road
Austin, TX 78758
512-838-3678

John Darragh
Electronic Systems Group
TRW
One Space Park
MS R10/2342
Redondo Beach, CA 90278
310-814-5919
darragh@desperado.etdesg.trw.com

Raffi Dionysian
UCLA
12550 Allin Street
Los Angeles, CA 90066
310-825-2266
raffi@cs.ucla.edu

Sam Dolinar
Mail Stop 238-420
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109
818-354-7403
sam@rtop71.jpl.nasa.gov

Melanie Dutkiewicz
MacDonald Dettwiler
13800 Commerce Parkway
Richmond, BC V6V 2J3
CANADA
278-3411 x2520
mel@mda.ca

Laura Ekroot
MS 238-420
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109
818-354-1280
laura@galois.jpl.nasa.gov

A. F. Ellman
Jet Propulsion Laboraotry
MS 525-3650
4800 Oak Grove Drive
Pasadena, CA 91109
818-306-6220

Robert Finch
Electrical Engineering Dept.
South Dakota State University
Box 2220
Brookings, SD 57007
605-688-5217
ee03@sdsumus.bitnet

Immanuel Freedman
Hughes STX Corporation
7601 Ora Glen Drive, #100
Greenbelt, MD 20770
301-513-7700
freedman@csbed.gsfc.nasa.gov

Tim Gerchar
Exabyte Corporation
1685 38th Street
Boulder, CO 80301
303-447-7366

Daniel Glover
Mail Stop 54-2
NASA/Lewis Research Center
21000 Brook Park Road
Cleveland, OH 44135
216-443-2847
caglove@lims01.lerc.nasa.gov

Ratan Guha
Dept. of Computer Science
University of Central Florida
Orlando, FL 32816
407-823-2341
guha@cs.ucf.edu

Amit Gulati
University of California - Santa Cruz
2015 Maciel Avenue
Santa Cruz, CA 95062
408-462-4344
amitg@cse.ucse.edu

Joseph Guzek
Hughes Aircraft Co.
7375 Executive Place
Suite 401
Seabrook, MD 20706
301-805-0345

Richard Haines
FCCD
NASA/Ames Research Center
N 269-4
Moffett Field, CA 94022
415-604-3373

Eric D. Halter
Naval Surface Warfare Center
Code K43
Dahlgren, VA 22448-5000
703-663-8542

Ray Hamel
Computer Science Dept.
Eastern Washington University
#86
Cheney, WA 99004
509-458-6260
rhamel@ewuvms.bitnet

Ingvil Hovig
University of Oslo
P. O. Box 1059
Blinderen 0316
OSLO
2-45-34-94
2-45-37-30 (Fax)
ingvil@usit.uio.no

Paul Howard
Dept. of Computer Science
Brown University
Providence, RI 02912-1910
401-863-7672
401-863-7657 (Fax)
pgh@cs.brown.edu

Kris Huber
Electrical Engineering Dept.
Utah State University
Logan, UT 84322-4120
801-750-3481
kris@bert.ee.usu.edu

Stephen Jones
Engineering Research Center for CFS
P. O. Box 6176
Mississippi State, MS 39762
601-325-2094
sjones@erc.msstate.edu

Ilia Kaufman
KCSL
5160 Yonge Street
Suite 1012
North York, Ontario M2N 6L9
CANADA
416-222-6112

Masoud Khansari
Dept. of Electrical Engineering
University of Toronto
10 Kings College Road
Toronto, Ontario
CANADA
416-978-6876
khansar@comsun.comm.toronto.edu

Michael Kirschner
Compression Research Group, Inc.
533 1/2 Diamond Street
San Francisco, CA 94114
415-641-1496
obnoid@well.sf.ca.us

Frederick Knops
Code MUS
NASA/Headquarters
600 Independence Avenue, SW
Washington, DC 20546
202-453-1182

Jim Korfanta
MS 126-239
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109
818-354-6599

Glen Langdon
Computer Engineering
University of California
Santa Cruz, CA 95064
408-459-2212
langdon@cse.ucsc.edu

C. C. Lu
Dept. of Math & Computer Science
Kent State University
Kent, OH 44242
216-672-2195
lucc@mcs.kent.edu

Matthew Maa
Electronic Data systems R&D
3551 Hamlin Road
4th Floor
Auburn Hills, MI 48326
313-370-1612
maa@edsdrd.eds.com

Manohar Mareboyana
Goddard Space Flight Center
USRA
Mail Code 930.1
Greenbelt, MD 20771
301-286-3397
manohar@chrpalg.gsfc.nasa.gov

Vishal Markandey
Texas Instruments
P. O. Box 655474
MS 238
Dallas, TX 75265
214-995-5513
markandey@ti.csc.com

Tassos Markas
Computer Science Dept.
Duke University
Durham, NC 27706
919-660-6564
am@cs.duke.edu

Nasir Memon
University of Nebraska - Lincoln
114 Ferguson Hall
Lincoln, NE 68588-0115
402-472-3485
memon@cse.unl.edu

Frederick Mertz
Photon Research Associates
9393 Towne Center Drive
San Deigo, CA 92121

Kristo Miettinen
General Electrical
259 Stone Ridge Drive
Jeffersonville, PA 19403
215-354-5713

Zavie Miller
ICE Centre, Environment Canada
373 Sussex Drive
3rd Floor, "E"
Ottawa, Canada K1A OH3
CANADA
613-996-4665
zmiller@cid.aes.doe.ca

Robert Moorhead
ERC for Comp. Field Sim.
Engineering Research Center
P. O. Box 6176
Mississippi State, MS 39762
601-325-2850
rjm@erc.msstate.edu

Amar Mukherjee
Professor
Dept. of Computer Science
University of Central Florida
Orlando, FL 32816
407-823-2763
amer@cs.ucf.edu

Michitaka Nami
Stanford University
3875 Park Blvd.
#B22
Palo Alto, CA 94306
415-494-2944
nami@isl.stanford.edu

Prakash Narayan
System Research Center
University of Maryland
College Park, MD 20742
301-405-3661
prakash@src.umd.edu

Jeffrey Newcomer
Hughes STX Corporation
4400 Forbes Blvd.
Lanham, MD 20706
301-286-7858
newcomer@ltp.gsfc.nasa.gov

Jeff Niehaus
Pixel Semiconducter, Inc.
3460 Lotus Drive
Plano, TX 75075
214-985-2230
niehaus@pixel.cirrus.com

Krystyna Ohnesorge
Computer Science Dept.
University of Zurich
Multimedia Laboratory
Winterthurerstrasse 190
Zurich CH-8057
SWITZERLAND
41-1-257-43-46
ohnesorge@ifi.unizh.ch

Naoya Ohta
MIT Media Lab
NEC Corporation
20 Ames Street
Room E15-384A
Cambridge, MA 02139
617-253-0380
nohta@media-lab.media.mit.edu

Arvind Patel
IBM Corporation
MS H18/025
5600 Cottle Road
San Jose, CA 95119
408-578-5267
408-256-5216 (Fax)

Michael Piotrowski
Eastern Washington University
E13506 Farwell Road
Spokane, WA 99207
509-928-9128

Fabrizio Pollara
Code 238-420
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109
818-354-4287
fabrizio@gabis.jpl.nasa.gov

Timo Raita
University of Turku
Lemminkaisenkatu 14A
SF-20520 Turku
FINLAND
raita@euroni.cs.utu.fi

Abbas Razavi
Zoran Corp.
1705 Wyatt Drive
Santa Clara, CA 95054

Todd Reed
Dept. of Elec, Eng. & Comp. Science
University of California
Davis, CA 95616-8553
916-752-4720
916-752-8428 (Fax)
trreed@ucdavis.edu

Robert Renner
Systems Development Division
TRW
One Space Park
E1/1063
Redondo Beach, CA 90278
310-812-4868
renner@spf.trw.com

Ashok Ian Roy
University of Central Florida
1800 Woodlake Drive
No. 201
Palm Bay, FL 32905
407-242-5092
roy@eola.cs.ucf.edu

Ilan Sadeh
School of Math
Tel Aviv University
Ramat-Aviv, Tel-Aviv 69978
ISRAEL
972-3-6408849
sade@math.tau.ac.il

Eric Schell
Video Telecom
1901 W. Braker Lane
Austin, TX 78758
512-838-9839

James Schiess
NASA/Langley Research Center
M.S. 157D
Hampton, VA 23665-5225
804-864-6718
schiess@eagle.larc.nasa.gov

Edward Seiler
Hughes STX Corporation
4400 Forbes Blvd.
Lanham, MD 20706
301-794-5404
seiler@amarna.gsfc.nasa.gov

Alan Sloan
President
Iterated Systems-Technoloy
5550 Peachtree Parkway
Norcross, GA 30092
404-840-0728
404-840-0029 (Fax)

David Stoft
Dept. of Computer Science
Western Washington University
Bellingham, WA 98225
206-671-3596
stoft@henson.cc.wwu.edu

James Storer
Computer Science Dept.
Brandeis University
Waltham, MA 02254
617-736-2714
617-736-2741 (Fax)
storer@cs.brandeis.edu

John Stutz
NASA/Ames Research Center
MS 269-2
Moffet Field, CA 94035

Stephen Tate
Dept. of Computer Science
Duke University
Durham, NC 27706
919-660-6557
srt@cs.duke.edu

Herb Taylor
MTS
David Sarnoff Research Center
CN5300
Princeton, NJ
609-734-2733
herb@apollo.sarnoff.com

James Tilton
NASA/Goddard Space Flight Center
Code 930.1
Greenbelt, MD 20771
301-286-9510
301-286-3221 (Fax)
tilton@chrpisis.gsfc.nasa.gov

Ming-Yuan Ting
University of Washington
5212 University Way, NE, Apt. #205
Seattle, WA 98105
206-526-0641
myting@uw-isdl.ee.washington.edu

Martin Turner
New Museums Site
University of Cambridge - Computer Lab
Pembroke St
403 Austin Bldg.
Cambridge CB2 3QG
UNITED KINGDOM
0223-334452
mit@uk.ac.cam.cl

Val Vaughn
The Aerospace Corp.
2350 E. El Segundo Blvd.
MS M4 1902
El Segundo, CA
310-336-1126
vaughn@aerospace.aero.org

Jeffrey Vitter
Computer Science Dept.
Brown University
Providence, RI 02912
401-863-7646
jsv@vs.brown.edu

Robert Vuolo
Jet Propulsion Lab
11983 Sentinel Pt.
Reston, VA 22091
703-620-0029

Ren-Yuh Wang
University of Washington
4240 8th Ave., NE
Apt. #301
Seattle, WA 98105
206-547-1336
wry@esdl.ee.washington.edu

Yao Wang
Polytechnic University
6 Metrotech Center
Brooklyn, NY 11201
718-260-3469
yaorainbow.poly.edu

Richard L. White
Space Telescope Science Institute
3700 San Martin Drive
Baltimore, MD 21218

Kel Winters
Advanced Hardware Architectures
P. O. Box 9669
Moscow, ID 83843
208-883-8000
kel@aha.com

Pen-Shu Yeh
NASA/Goddard Space Flight Center
Code 728
Greenbelt, MD 20771
301-286-4477
psyeh@psy.gsfc.nasa.gov

Yosi Yomdin
IGP Research & Development Ltd.
Weizman Science Park
P. O. Box 2435
Rehovot 76123
ISRAEL
972-8-469479

Ahmad Zandi
Computer Engineering
University of California-Santa Cruz
Santa Cruz, CA 95064
408-458-6670
zandi@ce.ucsc.edu

Marc Zipstein
I.G.M.
15 Av. Laumiere
75019 Paris
FRANCE
44-272842
zipstein@litp.ibp.fr

Li-He Zou
Dept. of Electrical Engineering
Louisiana Tech University
Ruston, LA 71272-0046
318-257-4301
zou@latech.edu

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1993	Conference Publication

4. TITLE AND SUBTITLE

Space and Earth Science Data Compression Workshop

5. FUNDING NUMBERS

936

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

93B00008

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

NASA CP-3183

11. SUPPLEMENTARY NOTES

This workshop was organized and sponsored by the National Aeronautics and Space Administration (NASA), and cosponsored by the IEEE Computer Society Technical Committee on Computer Communications (TCCC). It was held in conjunction with the 1992 Data Compression Conference (DCC '92), which was cosponsored by the IEEE TCCC in cooperation with NASA and the Center of Excellence in Space Data and Information Sciences at Goddard Space Flight Center.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category 59

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This document is the proceedings from a "Space and Earth Science Data Compression Workshop," which was held on March 27, 1992, at the Snowbird Conference Center in Snowbird, Utah. This workshop was held in conjunction with the 1992 Data Compression Conference (DCC '92), which was held at the same location, March 24-26, 1992. The workshop explored opportunities for data compression to enhance the collection and analysis of space and Earth science data. The workshop consisted of eleven papers presented in four sessions. These papers describe research that is integrated into, or has the potential of being integrated into, a particular space and/or Earth science data information system. Presenters were encouraged to take into account the scientists's data requirements, and the constraints imposed by the data collection, transmission, distribution, and archival system. The workshop was organized by James C. Tilton of the Goddard Space Flight Center, and Sam Dolinar of the Jet Propulsion Laboratory.

14. SUBJECT TERMS

Data Compression, Image Compression, Signal Processing, Image Processing, Space Science, Earth Science

15. NUMBER OF PAGES

156

16. PRICE CODE

A08

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

NASA

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