SANDIA REPORT

SAND2005-3402
Unlimited Release
Printed July 2005

Correlation and Image Compression for Limited-Bandwidth CCD

Douglas G. Thompson

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.doe.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/ordering.htm

SAND2005-3402
Unlimited Release
Printed July 2005

Correlation and Image Compression for Limited-Bandwidth CCD

Douglas G. Thompson
SAR Applications Department

Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-0519

ABSTRACT

As radars move to Unmanned Aerial Vehicles with limited-bandwidth data downlinks, the amount of data stored and transmitted with each image becomes more significant. This document gives the results of a study to determine the effect of lossy compression in the image magnitude and phase on Coherent Change Detection (CCD). We examine 44 lossy compression types, plus lossless zlib compression, and test each compression method with over 600 CCD image pairs. We also derive theoretical predictions for the correlation for most of these compression schemes, which compare favorably with the experimental results. We recommend image transmission formats for limited-bandwidth programs having various requirements for CCD, including programs which cannot allow performance degradation and those which have stricter bandwidth requirements at the expense of CCD performance.

ACKNOWLEDGEMENTS

This work was funded by the US DOE NNSA/NA-22 Office of Nonproliferation & National Security, Office of Research and Development, under the Advanced Radar System (ARS) project.

Special thanks to Doug Bickel and John Delaurentis for helping to review this work and to get me through the sticky mathematics.

CONTENTS

This page left intentionally blank.

1 Introduction

As radars move to Unmanned Aerial Vehicles with limited-bandwidth data downlinks, the amount of data stored and transmitted with each image becomes more significant. This document gives the results of a study to determine the effect of lossy compression in the image magnitude and phase on Coherent Change Detection (CCD). We examine several compression types, the primary method being to simply remove bits from the data after taking the Nth root of the magnitude. We also examine the effect on data size of using lossless zlib compression on the data resulting from the lossy compression.

2 Data Compression Methods Tested

This section discusses the data compression methods tested in this study.

2.1 Lossy Data Compression Methods

This section discusses the lossy data compression methods considered. Table 1 and Table 2 provide a summary of the different methods tried, from two different points of view. Table 1 shows the different trials sorted by relative output size. For the magnitude data, we perform the indicated operation, then take the N least significant bits, where N is the number in the second column of the table. Any values that do not fit into N bits are hard-limited to 2^N-1. For JPEG cases, we take the square root of the magnitude and quantize to form 8-bit data, then JPEG-compress this data with the quality factor indicated. Those cases where the operation is indicated as Lloyd refer to quantization Method I as described in [1] and summarized below. Those cases indicated as FFT refer to quantizing the Fourier Transform of the image instead of the image itself, as discussed in [2] and summarized below.

For FFT cases, we quantize the I and Q components of the image in the frequency domain. In all other cases, the operation only refers to the magnitude data. The phase data is simply quantized to the indicated number of bits, keeping the most-significant bits.

Unlike the uniform quantization used in all other cases, the cases using Lloyd's quantizer map arbitrary sets of input values into an arbitrary set of output quanta. The compressor maps all input values that lie in the first set to the integer label "0", the values that lie in the second set to the label "1", and so on. Only the output quanta and the labels are stored or transmitted with the image. The decompressor then maps each of the labels to the appropriate output quantum. We implemented this quantization scheme on the square root of the magnitude data. We first calculate the optimum input ranges and the corresponding optimum output quanta using Lloyd's Method I, using a histogram of the data as an approximation to the probability distribution function. We then map the input values into the integer labels. Note that this method requires the transmission of the 2^{Nm} quanta (floating-point values) in addition to the image data, where N_m is the number of magnitude bits. We have ignored these few extra data words in our calculation of compression ratios. The process of calculating the input regions and quanta and then quantizing the data takes prohibitively long in Matlab but runs fairly fast in C, so we implemented this compression scheme in C.

For the FFT cases, we take the two-dimensional FFT of the complex image to yield the image domain. We assume that the image was a two-dimensional FFT of some windowed phase histories. While this is not entirely accurate for images produced by OSA, it makes a good approximation. We multiply by the inverse of a Taylor window in both range and azimuth to obtain a data set with a fairly uniform distribution in I and Q. We then quantize the I and Q parts of this data. We tried cases with uniform quantization whose limits were defined by the maximum absolute value of the data ("full range") and cases where the limits were defined by 1, 2, 3, and 4 standard deviations of the data. Since the I and Q parts are nearly identically distributed and uniform, we only did cases where the I and Q parts are quantized to the same

number of bits. The decompressor combines I and Q parts to make complex data, multiplies by a Taylor window in two dimensions, then takes the two-dimensional inverse FFT.

Table 1. The lossy data compression methods used in this study and the relative output size.

Operation	# Magnitude Bits kept	# Phase bits kept	Relative size
None	16	16	1
Square root	8	12	0.625
Square root	8	8	0.5
Square root	8	6	0.438
Divide by 2	9	4	0.406
Divide by 2	8	4	0.375
Divide by 4	8	4	0.375
Square root	8	4	0.375
FFT, full range	6	6	0.375
Square root	6	6	0.375
Divide by 8	7	4	0.344
Square root	5	6	0.344
Square root	8	2	0.313
Square root	6	4	0.313
Square root	4	6	0.313
Third root	4	6	0.313
95% JPEG	8	8	0.30
Lloyd	6	3	0.281
Square root	5	4	0.281
90% JPEG	8	8	0.28
80% JPEG	8	8	0.27
Square root	6	2	0.25
Lloyd	5	3	0.25
Square root	5	3	0.25
FFT, 1 std	4	4	0.25
FFT, 2 stds	4	4	0.25
FFT, 3 stds	4	4	0.25
FFT, 4 stds	4	4	0.25
FFT, full range	4	4	0.25
Lloyd	4	4	0.25
Square root	4	4	0.25
Third root	4	4	0.25
Fourth root	3	5	0.25
Lloyd	3	5	0.25
Natural log	3	5	0.25
95% JPEG	8	6	0.24
90% JPEG	8	6	0.22
Third root	4	3	0.219
Fourth root	3	4	0.219
Natural Log	3	4	0.219
Natural log	3	3	0.188
95% JPEG	8	4	0.178
Fourth root	3	2	0.156
Fourth root	2	1	0.094

Figure 1 shows histograms of a typical clutter image. Urban images would be expected to be approximately the same but with bright pixels occurring more often. Note that the most significant bits of the magnitude are not used very often, so chopping off a few bits on this end will have little effect. Examining these plots closely, we can see two issues in lowering the bit-count of our image representation: quantization and clipping. In almost any operation where we reduce the number of bits used to store a number, we introduce quantization error. If we start with a 16-bit number, we know that its square root can be stored in 8 bits, but with the loss of a fractional part, or in other words, with quantization error introduced. The other issue is clipping– to store a number in N bits, we must limit all values greater than or equal to 2^N by setting them equal to 2^N-1. If we desire to store the square root of our 16-bit magnitude in only 4 bits, we can see from the histogram that ~1.0% of the pixels in this particular image are greater than 2^N-1. At this point, we see this as a reasonable trade-off to consider making to reduce the data rate.

As shown in the last plot in Figure 1, the phase has a uniform distribution across the entire 16-bit domain (representing 0 to 2π). This tells us that we can get the best gain in data rate reduction for the quality degradation simply by dropping the least significant bits. We will see another significant quality of the phase in the following section: since it is uniformly distributed across its entire domain, it rarely sees gains due to lossless compression when it is compressed by itself.

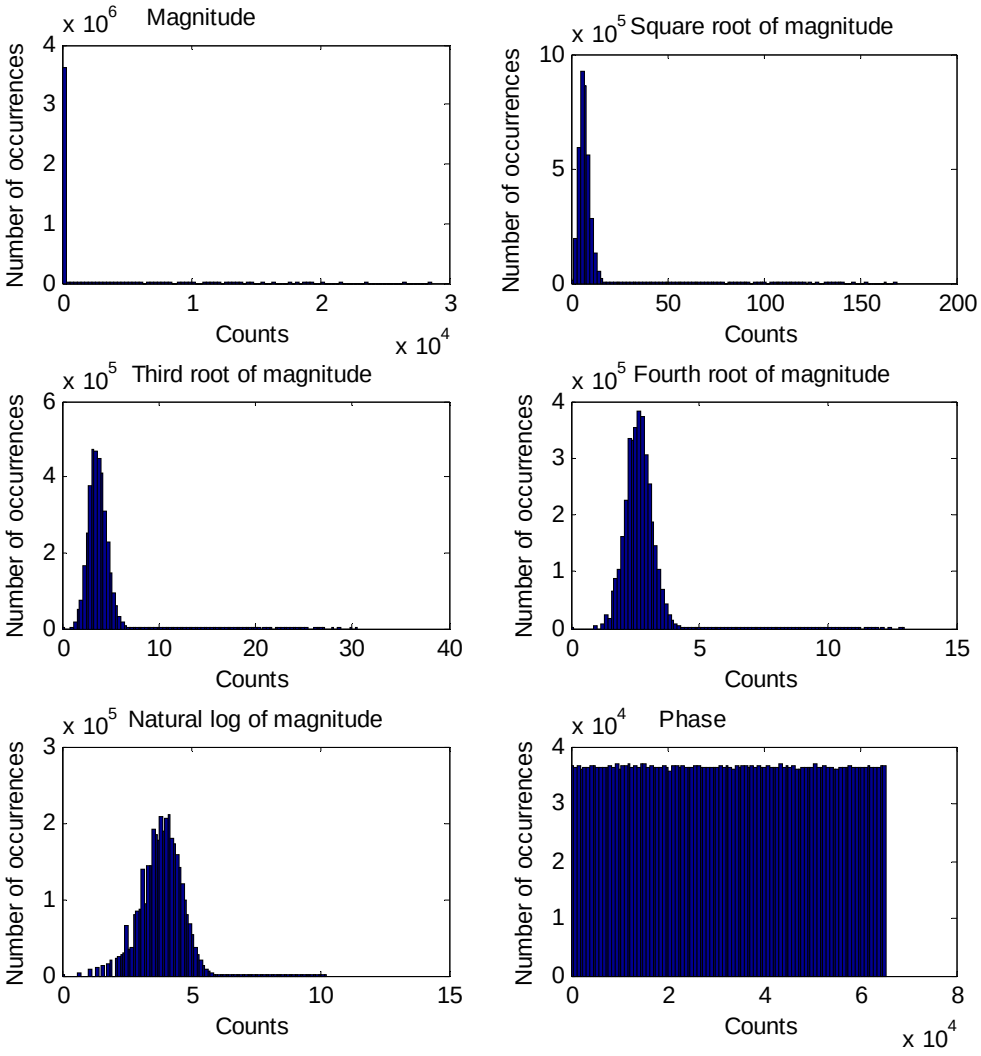

Figure 1. Histograms of a typical clutter image.

Table 2 shows a matrix view of the lossy compression methods in this study. The top row shows the number of phase bits. The leftmost two columns show the number of magnitude bits and the nonlinear operation performed. The operation column indicates the power of the root for a root operation, "Log" for the natural log operation, the quality factor (indicated by a "%" symbol) for JPEG compression, "Lloyd" for Lloyd's quantizer, and "FFT" for images quantized in the frequency domain. We then indicate in matrix form which combinations were tested. The table also includes the results of the tests at the highest level, indicating those combinations which resulted in no noticeable degradation ("Good"), those which resulted in Change Detection not working at all ("Bad"), and those with significant degradation that may still be useful for various applications ("X"). The details of the results are shown in later sections.

Table 2. A matrix view of the lossy compression methods examined in this study. Entries indicated "good" have no noticeable degradation; those marked "Bad" have essentially no information remaining, and those marked "X" have some significant degradation.

Mag bits⇓	Phase bits ⇒ / ⇓Op	16	12	8	6	5	4	3	2	1
16	1	Good								
8	2		Good	Good	Good		X		Bad	
6					X		X		Bad	
5					X		X	X		
4					X		X			
4	3				X		X	X		
3	4					X	X		Bad	
2										Bad
8	80%			X						
8	90%			X	X					
8	95%			X	X		X			
6	FFT				X					
4							X			
6	Lloyd							X		
5								X		
4							X			
3						X				
3	Log					X	X	X		
9	/2						X			
8							X			
8	/4						X			
7	/8						X			

2.2 Lossless Data Compression Methods

In addition to the lossy data compression discussed above, these tests examined the effects of lossless data packing and compression methods on the resulting size and on CPU requirements. We first pack the bits so that all bits of each byte are used, then we further compress them using the lossless zlib compression algorithm.

For these tests, we wrote a simple C function that can be called from Matlab that packs data with an arbitrary number of bits (anything from 1 bit per item to 32 bits per item) into a stream of standard 8-bit bytes, and a corresponding function that unpacks the data [3]. In addition to eliminating unused bits, these functions put the data into a near-optimum order giving maximum compression in the following step with a simple packing scheme. In particular, any whole-byte quantities are packed separately from the remaining partial-byte quantities. For example, if we are packing 12-bit data, all the least-significant bytes are put together first, followed by the most-significant 4 bits of each item, packed with multiple items in each byte. If we are packing 32-bit

data (e.g. the original GFF format), then we store the least-significant phase byte of all pixels, then the most-significant phase byte of all pixels, then the least-significant magnitude byte of all pixels, then the most-significant magnitude byte of all pixels. Such ordering significantly increases the compression attained by zlib in the following step. We tested the compression using these packing functions in two different ways. For one set of tests, we pack and store all the magnitude bits, then pack and store all the phase bits. For the other set of tests, we combine magnitude and phase into one item, and pack and store this aggregate item all together.

The packed data is then compressed using the zlib compression library [4]. This library includes compression schemes used by the Unix command gzip and by Windows .zip file software. This step reduces the data size by a factor of about 0.75-0.8 on average, depending on image contents. Note that the choice of zlib compression is near-optimal in the same sense as the bit-packing scheme described above: other schemes may result in better compression ratios but are not as simple to implement, and may take more CPU time.

For those cases where the magnitude is JPEG compressed, there is no point in attempting further compression on the magnitude. Since the phase is uniformly distributed, it can achieve very little if any compression by itself. Thus, for the JPEG cases, we do not attempt any lossless compression.

3 Compressed CCD Tests

We tested a total of 633 image pairs from two different flight campaigns. The majority of the image pairs (558 pairs) came from a single flight on November 1, 2004. During this flight, nine circle passes were flown around two sites while people and vehicles moved about on the ground. In each pass, the radar collected 31 images suitable for CCD processing. The first six passes imaged the first site. This allows 15 different combinations of passes per image at this site. The last three passes imaged a second nearby site, allowing three different combinations of passes per image.

The remainder of the image pairs (75 pairs) came from a series of three flights, on November 17 and 21, 2004 and January 7, 2005. Some of the pairs were within a single flight and others spanned flights. There was significant rain in the Albuquerque area between each of these pairs of flights, so any of the pairs which span flights had relatively poor CCD results. Part of the purpose of including these image pairs is to check the similarities and differences between the behavior of highly correlated image pairs versus less-correlated image pairs in the presence of lossy compression.

For some of this data, we ran two different CCD processors. The first was the GEARSDriver.exe software provided by GA. This is a command-line CCD processor with a processing engine similar to that in CLAW. The input is in the form of Lynx Image Files (.limg), and the output is TIFF files (.tif), scaled so that a pixel value of 255 represents a correlation of 1. We are not privy to the source code for this processor and thus do not know and cannot change the algorithms or parameters used. This processor was used to process all of the pairs described above. The second CCD processor was a Matlab script written by Armin Doerry and processed only a subset of the 558 pairs from the November 1 flight. This script takes in two complex floating-point matrices and corresponding header structures, and creates a floating-point matrix as the output CCD product. The registration is done using Matlab functions provided as part of the Image Processing Toolbox. The GEARSDriver showed registration problems much less often than the simple Matlab script.

Both CCD processors had a few cases where the correlation went up as the images were degraded. Since we expect degradation in the images to produce degradation in the CCD product, we threw out all such cases, attributing them to bad registration in the un-degraded case. We also threw out cases where the registration shifted the images so much that the resulting CCD image was smaller in either dimension than 1600 pixels. This was both for convenience (we chose a 1600x1600-pixel region of each image to analyze) and because cases where the images were shifted more than this are likely to have inaccurate registration. Finally, we threw out cases where there were no 30x30 blocks whose minimum correlation was greater than 0.7 (since this is required for our measure of contrast between bright and dark pixels; see below). From the November 1, 2004 data set, 17 pairs were thrown out because of increasing correlation. 22 pairs were thrown out because there were no blocks with minimum correlation greater than 0.7, and 519 pairs were used.

We need a criterion or set of criteria for determining whether CCD still works given a particular method of lossy compression. The ideal criterion would simply be the ability to detect changes.

This is difficult to measure directly, since we have no a priori knowledge of which pixels contain changes and which do not. Therefore, we measured several different quantities as approximations to the ideal criterion. Perhaps the best picture of the performance of a CCD algorithm in the presence of lossy compression can be obtained by examining several or all of these parameters; therefore, we describe and present the data for all of them in this document.

The first measure of CCD performance is simply to calculate the average correlation across the image. While this is not especially useful for comparing images because of the many factors which go into correlation measurements (like actual changes in the scene, thermal noise, navigation errors, etc.), it is more interesting if we compare the average correlation for a degraded image pair to the average correlation for the original un-degraded version of the same CCD product. The correlation in a degraded image pair divided by the correlation in the original image pair should be the correlation due to lossy compression, as shown by the following equations, using the typical model for correlation.

$$\gamma_{original} = \gamma_{temporal} \cdot \gamma_{thermal} \cdot \gamma_{geometry} \cdot \ldots \tag{1}$$

$$\gamma_{compressed} = \gamma_{compression} \cdot \gamma_{temporal} \cdot \gamma_{thermal} \cdot \gamma_{geometry} \cdot \ldots = \gamma_{compression} \cdot \gamma_{original} \tag{2}$$

In the typical CCD problem, the value to be derived is $\gamma_{temporal}$, the correlation due to changes in the scene over time. For our current purposes, we desire to find the reduction in correlation due to the compression, or

$$\gamma_{compression} = \frac{\gamma_{compressed}}{\gamma_{original}}. \tag{3}$$

We examine both the compression itself and its equivalent SNR, calculated as

$$SNR_{equivalent,\rho} = 10 \cdot \log_{10}\left(\frac{\gamma}{1-\gamma}\right). \tag{4}$$

In order to calculate the equivalent SNR degradation due to the lossy compression, we apply Equation (4) to the full correlation, then subtract the values for the degraded CCD products from those for the original CCD products.

The second measure of CCD performance is to calculate the average correlation across those regions of the image with high correlation in the original CCD product. It is not a particularly bad thing if the dark regions (indicating change) of a CCD product get darker (more strongly indicating change), so we desire to leave them out of the calculation. In calculating this parameter, we divide the CCD image into 30x30-pixel blocks, and find all those blocks whose minimum correlation is greater than 0.7 in the original CCD product. We find the average correlation over these same blocks in the degraded CCD product. Again, we examine this parameter in linear correlation space relative to the original value, and as an equivalent SNR degradation.

The third measure of CCD performance is to calculate the difference in dB between the equivalent SNR of the average bright pixel and the average dark pixel. For this calculation, bright pixels are defined as in the previous paragraph as those 30x30-pixel blocks whose minimum correlation in the original CCD image is greater than 0.7. Dark pixels are defined as those 30x30-pixel blocks whose maximum correlation in the original CCD image is less than 0.3. If the dark pixels really are caused by changes in the scene as we assume, then we desire to maximize the difference between the bright and dark pixels so that we can easily pick out the dark ones.

The fourth measure of CCD performance is the RMS of the difference in pixel values between the original and degraded CCD images. This measurement is more sensitive to changes in the individual pixel values, compared to the above three measures which examine changes in averages over relatively large numbers of pixels.

4 Test Results

This section examines in detail the results of the tests described above.

4.1 Subjective Image and CCD Quality

In addition to objectively measuring the CCD results with the degraded images, we examined the images for subjective viewing quality. Both the degraded magnitude image and the degraded CCD result were examined qualitatively.

To check the magnitude image quality, we convert the quantized values back to magnitude (undoing the non-linear operations performed, but not undoing the lossy effects of quantization or JPEG compression), then take the square root to obtain a quarter-power image. We multiply the resulting value by 8, and then use it to index a linear grayscale look-up table. This is similar to one of the common ways of displaying the GFF images: to take the square root of the original 16-bit magnitude data, and then multiply by 8 to give an 8-bit grayscale index. We then save these images as 100%-quality JPEGs, since this is a convenient image format known to preserve image appearance fairly well, as the human eye can measure it. Note that only the operations and number of bits stored for the magnitude part of the image affect this result, since the phase is thrown away in creating the magnitude image.

Not surprisingly, we could see no visible difference between the original image and any of the images where the operation was a square root and at least 5 magnitude bits were saved. The net effect of all the operations performed in these cases was the same as for the original image. Multiplying the quarter-power data by 8 and then limiting the resulting values to fit into 8-bit quantities is equivalent to limiting the original quarter-power data to 5 bits. Quantizing the square root of the magnitude is done in either case.

Most of the cases where we took a different operation than the square root and/or kept fewer than 5 bits look very similar to the original image. When rapidly changing from one case to another on the computer screen, the eye catches the changes so that it is obvious that the two images are not identical. However, it is hard to discern any difference by looking at the original image next to the degraded image. Of those cases examined in this study, the only exceptions to this statement are the 4-bit square-root case, and the 2-bit fourth-root case. In both of these cases, too few bits are kept to allow the preservation of dynamic range in the image. Referring back to the histograms in Figure 1, it is the presence of a few bright points in the image that the eye sees as contrast. With the square-root case, four bits limits the data to so that not all of the primary hump is represented, and none of the upper tail. The image looks very gray and uninteresting. If we scale the image so that we get some bright points, then too much of the histogram becomes bright as well, and the image is washed out.

A few sample cases of a SAR image of Sandia's radar calibration range are shown in Figure 2. Since the images are displayed as 5-bit square-root of magnitude images, none of the square-root cases with 5 or more bits is shown. First we see the original image, with three corner reflectors visible. Next is the 4-bit square-root of magnitude image. Notice the dim corner reflectors. If we had chosen a more interesting scene, we would notice that all image features are similarly

dim. The bottom row shows 4-bit cube root of magnitude and 3-bit natural log of magnitude. For the eye, it is difficult to detect any differences between these images and the original.

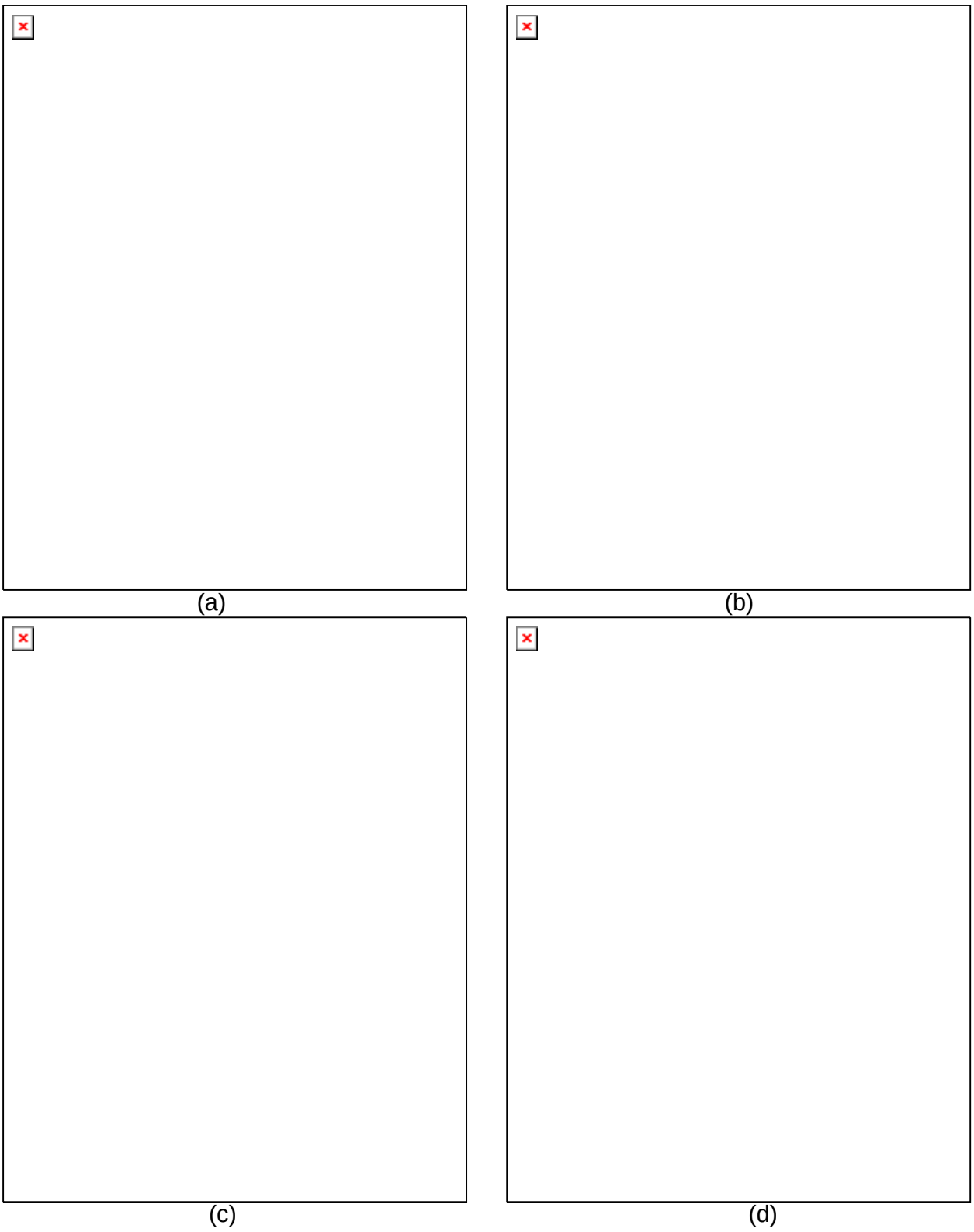

Figure 2. Sample SAR image showing degradation: (a) original, 16-bit magnitude image (displayed as 5-bit square root) (b) 4-bit square-root of magnitude, (c) 4-bit cube root of magnitude, (d) 3-bit natural log of magnitude.

As discussed above, most of the CCD processing for this study was done by the GEARSDriver software provided by GA. The CCD output products produced by this software are in TIFF format, so we simply examine these images for the qualitative check of CCD results.

In general, the square root operation produces a CCD product that better approximates the original than other operations. Those cases with 8-bit square root of magnitude data and 6 bits or more of phase data have performance nearly indistinguishable from the original. Those cases using the square root of magnitude with at least 4 bits of magnitude data and at least 4 bits of phase data are only slightly darker on the whole, but bright points tend to lose correlation.

Particularly among the cases with 8 total bits, the performance seems to be optimized by balancing the number of bits between phase and magnitude. In all cases where significant clipping occurs, the correlation on and around bright points in the image (those affected by clipping) degrades noticeably. All trials with 2 bits of phase data or fewer are significantly degraded (enough that they are marked as “bad” in Table 2) and should not be used for CCD.

Figure 3 shows a few samples of the CCD product corresponding to the image in Figure 2. For this case, we show the original product, the 8-bit square-root of magnitude and 6-bit phase case (the product this paper recommends as best compression with performance equal to the original), and two cases with 8 total bits. Notice that with 4 bits of square-root of magnitude and 4 bits of phase, the average correlation remains high, but the correlation around bright points is noticeably lower, due to the magnitude limiting seen above in the SAR images.

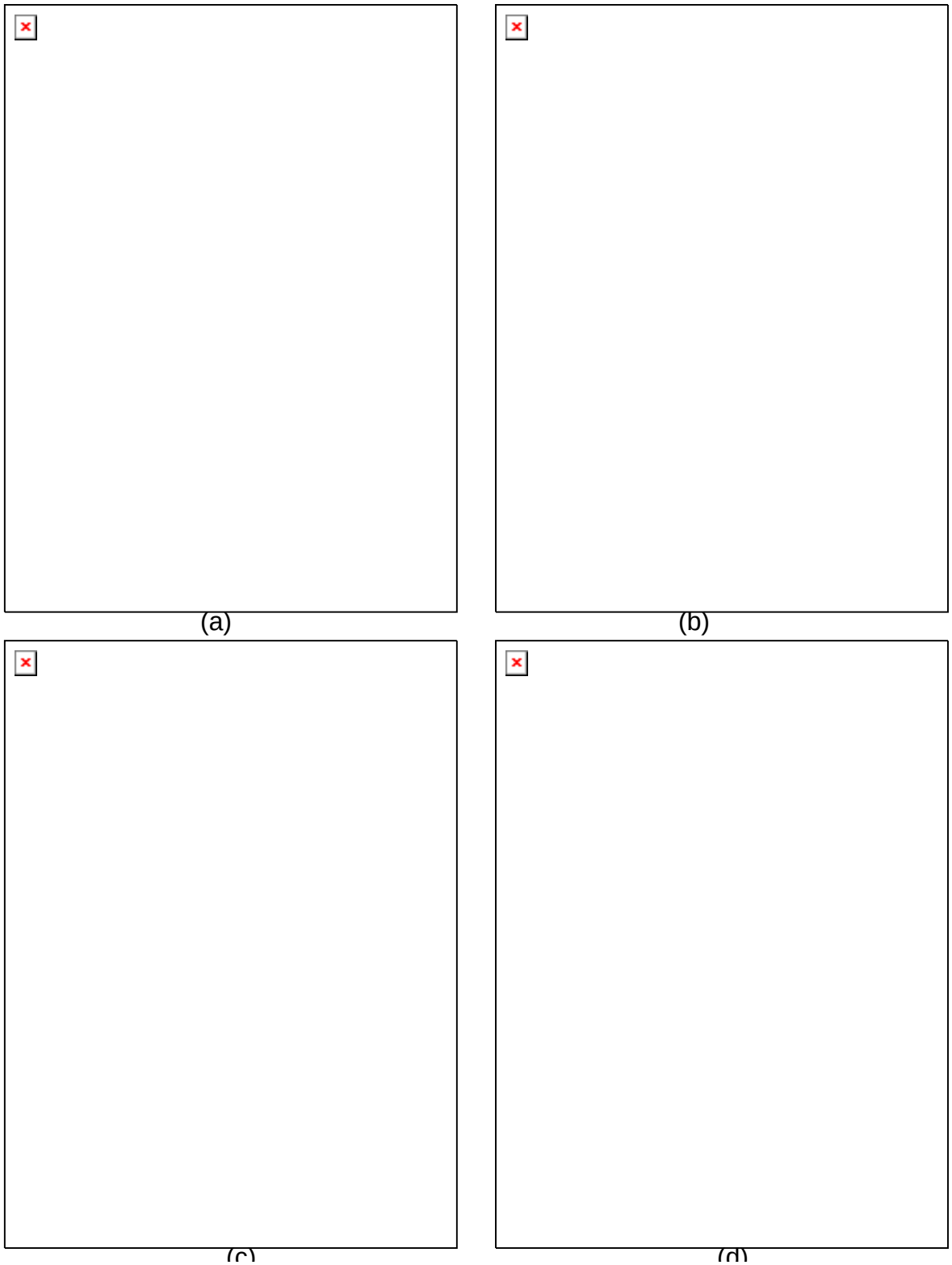

Figure 3. Sample CCD images from Sandia's radar calibration range: (a) original 16-bit magnitude, 16-bit phase, (b) 8-bit square root of magnitude, 6-bit phase, (c) 4-bit square root of magnitude, 4-bit phase, and (d) 4-bit cube root of magnitude, 4-bit phase

4.2 Compression Results

This section discusses the amount of compression obtained from the combination of the various lossy compression methods and lossless zlib compression. Table 3 summarizes this data. For most cases, the lossy compression ratio given is simply the number of magnitude bits plus the number of phase bits in the degraded product, divided by 32, the total number of bits in the initial product. For the JPEG cases, the lossy compression ratio is the number of phase bits in the degraded product divided by the number of phase bits in the original, plus the ratio of bytes in the magnitude JPEG to the bytes in the original magnitude data. The two columns labeled as the zlib compression ratio give the lossless compression ratio for the data with the two different types of data packing. This ratio is 1 for all JPEG cases because zlib is not used in these cases. As discussed in Section 2.2, the one packing method involves combining the integers representing magnitude with the integers representing phase before packing the data into 8-byte quantities. The combination is of the form

$$mag + phase \cdot 2^{N_{bits,magnitude}} . \qquad (5)$$

This method is indicated as "Mag & Phase together" in the table. The second method is to pack all the magnitude data into 8-byte quantities and pack all the phase data into 8-byte quantities, then concatenate the resulting arrays together. This is indicated as "Mag & Phase separate" in the table. On average, the case with phase and magnitude data separate compresses about 1% better than the case with the two items packed together, meaning that the compression ratio is better by 0.01. However, we see that for many cases the two numbers are identical or nearly so. For cases with 8-bit magnitude data, the packed data from the two methods is essentially if not exactly the same, so the compression ratios must of necessity be similar.

Table 3. Compression ratios with the two methods of packing data.

	Lossy Compression ratio	Zlib compression ratio / Phase, mag together	Total compression ratio / Phase, mag together	Zlib compression ratio / Phase, mag separate	Total compression ratio / Phase, mag separate
16-bit mag,16-bit phase	1.00	0.53	0.53	0.77	0.77
8-bit mag^(1/2),12-bit phase	0.63	0.68	0.43	0.81	0.51
8-bit mag^(1/2),8-bit phase	0.50	0.78	0.39	0.78	0.39
9-bit mag/2, 4-bit phase	0.41	0.93	0.38	0.87	0.35
8-bit mag/2, 4-bit phase	0.38	0.92	0.34	0.92	0.34
8-bit mag^(1/2),6-bit phase	0.44	0.75	0.33	0.75	0.33
8-bit mag/4, 4-bit phase	0.38	0.87	0.33	0.87	0.33
8-bit 95% jpeg,8-bit phase	0.32	1.00	0.32	1.00	0.32
5-bit mag^(1/2),6-bit phase	0.34	0.92	0.32	0.98	0.34
6-bit mag^(1/2),6-bit phase	0.38	0.84	0.32	0.92	0.34
7-bit mag/8, 4-bit phase	0.34	0.86	0.30	0.92	0.32
6,6-bit I/Q of FFT	0.38	0.78	0.29	0.91	0.34
8-bit 90% jpeg,8-bit phase	0.29	1.00	0.29	1.00	0.29
4-bit mag^(1/2),6-bit phase	0.31	0.91	0.28	0.91	0.28
8-bit 80% jpeg,8-bit phase	0.27	1.00	0.27	1.00	0.27
4-bit mag^(1/3),6-bit phase	0.31	0.84	0.26	0.85	0.27
8-bit mag^(1/2),4-bit phase	0.38	0.69	0.26	0.69	0.26
8-bit 95% jpeg,6-bit phase	0.26	1.00	0.26	1.00	0.26
6-bit mag^(1/2),4-bit phase	0.31	0.81	0.25	0.88	0.28
5-bit mag^(1/2),4-bit phase	0.28	0.90	0.25	0.95	0.27
4,4-bit I/Q FFT, 2 stds	0.25	0.97	0.24	0.96	0.24

3-bit Lloyd's, 5-bit phase	0.25	0.97	0.24	0.97	0.24
6-bit Lloyd's, 3-bit phase	0.28	0.86	0.24	0.92	0.26
4-bit Lloyd's, 4-bit phase	0.25	0.95	0.24	0.91	0.23
4,4-bit I/Q FFT, 1 stds	0.25	0.93	0.23	0.93	0.23
8-bit 90% jpeg,6-bit phase	0.23	1.00	0.23	1.00	0.23
5-bit Lloyd's, 3-bit phase	0.25	0.92	0.23	0.96	0.24
4-bit mag^(1/2),4-bit phase	0.25	0.89	0.22	0.87	0.22
5-bit mag^(1/2),3-bit phase	0.25	0.89	0.22	0.95	0.24
4,4-bit I/Q FFT, 3 stds	0.25	0.87	0.22	0.87	0.22
3-bit ln mag,5-bit phase	0.25	0.86	0.22	0.90	0.22
3-bit mag^(1/4),4-bit phase	0.22	0.98	0.21	0.82	0.18
3-bit ln mag,4-bit phase	0.22	0.98	0.21	0.86	0.19
4-bit mag^(1/3),3-bit phase	0.22	0.96	0.21	0.77	0.17
3-bit mag^(1/4),5-bit phase	0.25	0.83	0.21	0.86	0.22
4-bit mag^(1/3),4-bit phase	0.25	0.81	0.20	0.79	0.20
4,4-bit I/Q FFT, 4 stds	0.25	0.78	0.20	0.78	0.19
8-bit 95% jpeg,4-bit phase	0.19	1.00	0.19	1.00	0.19
8-bit mag^(1/2),2-bit phase	0.31	0.62	0.19	0.62	0.19
6-bit mag^(1/2),2-bit phase	0.25	0.77	0.19	0.85	0.21
4,4-bit I/Q of FFT	0.25	0.69	0.17	0.69	0.17
3-bit ln mag,3-bit phase	0.19	0.91	0.17	0.85	0.16
3-bit mag^(1/4),2-bit phase	0.16	0.87	0.14	0.74	0.12
2-bit mag^(1/4),1-bit phase	0.09	0.58	0.05	0.50	0.05

4.3 Timing Benchmark Results

In addition to testing the CCD results obtained from images degraded by lossy compression, we ran some rough benchmarks to gauge how long the conversions, bit-packing, and compression take. These benchmarks were run on a Linux machine with dual 2.4 GHz Pentium-4 hyperthreaded processors. Since we did not optimize the code we are testing in these benchmarks, the results simply give a rough idea of the length of the various tasks.

Figure 4 compares a few parameters between the two byte-packing schemes described in Section 2.2. All parameters are average values plotted versus the lossy compression ratio. The data shown for the zlib compression ratio was shown in table form in the previous section and appears again here. The mean compression time is the time to form the JPEG image (in Matlab) for those cases using that compression method, and the lossless zlib compression time (in C) for all other cases. The mean packing time is the time it takes to combine the N-bit quantities into properly ordered sets of 8-bit bytes. For JPEG cases, this only involves the phase, and thus there is no difference between the two methods. For all other cases, the “Mag & phase separate” method requires two calls to a C function that packs the data into 8-bit quantities while the “Mag & Phase together” method requires one such call. The extra function call does not double the execution time since the function has less data to operate on each time, but it does increase the total execution time by an average of 0.4 seconds. The total average processing time to prepare the data for a low-bandwidth link is about 0.43 seconds smaller for the “Mag & Phase together” case. For this reason and because this method is simpler conceptually (especially for images using a total of 8 bits per pixel), for the rest of this document we assume that we will use the “Mag & Phase together” method. The compression data shown in Section 4.4 includes this assumption.

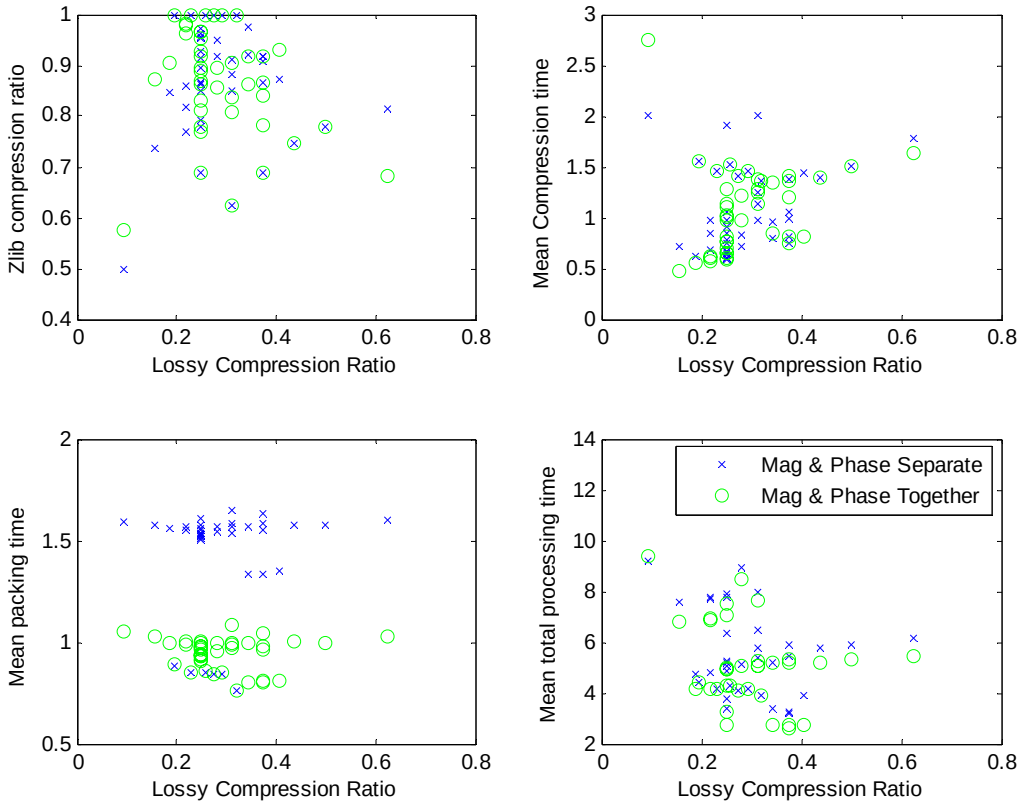

Figure 4. A few parameters comparing the two byte-packing methods. The legend in the bottom-right plot applies to all four plots.

Table 4 summarizes the timing benchmarks for all the lossy compression schemes tested. Note that there is a bit of a mix between C code and Matlab code and that no particular effort was made to optimize the execution speed of any of the functions tested. The column indicated as "bit-chopping time" refers to performing the indicated root or other operation on the original magnitude data and truncating this result and the phase to the indicated number of bits. This takes significantly longer for cube roots and fourth roots in Matlab. A program desiring to use one of these compression schemes would benefit from some type of lookup table or other approximation to these operations. Since we will truncate to an integer after the operation, it only needs to be accurate to the nearest integer anyway. The bit-chopping time for 16-bit mag, 16-bit phase (the original data) is artificially inflated because this test used the same code for all cases. The code could easily be optimized by removing such terms as $2^{16} / 2^{N}$ where N=16. We chose those areas of the test to use C code instead of Matlab scripts based on quick tests indicating that there would be a significant speedup. For example, we ran just a few trials of Lloyd's quantizer in Matlab and C and determined that C was many times faster. However, a similar set of trials on the cases using the square root and the third root indicated that the difference was insignificant. If we choose not to use zlib compression, then we would recommend a simpler and faster bit-packing scheme, especially for cases with multiples of 8 total bits.

We can see a few interesting features in this data. The FFT-based compression schemes take the longest by far. My implementation of Lloyd's quantizer is much faster for small numbers of bits

than for large numbers. As mentioned above, this implementation of the third-root and fourth-root schemes is much slower than the square-root schemes, but this difference could probably be eliminated in an optimized system. The zlib compression for 16-bit magnitude, 16-bit phase takes an extremely long time in some cases, distorting the mean. It is unknown why this happens but we have repeated this test several times with similar results. The most interesting point shown in this table is that the FFT-based methods probably take too long to be useful operationally, unless they give a much better performance than the other methods.

Table 4. A summary of the timing benchmarks.

	Lossy compression ratio	Zlib compression ratio	Total compression ratio	Mean compression time *	Mean packing time **	Mean bit-chopping time ***	Mean total processing time
16-bit mag,16-bit phase	1.00	0.53	0.53	31.31	0.89	0.92	33.12
8-bit mag^(1/2),12-bit phase	0.63	0.68	0.43	1.64	1.03	2.80	5.47
8-bit mag^(1/2),8-bit phase	0.50	0.78	0.39	1.50	0.99	2.81	5.30
9-bit mag/2, 4-bit phase	0.41	0.93	0.38	0.82	0.81	1.13	2.75
8-bit mag/2, 4-bit phase	0.38	0.92	0.34	0.82	0.81	1.11	2.74
8-bit mag^(1/2),6-bit phase	0.44	0.75	0.33	1.39	1.01	2.80	5.20
8-bit mag/4, 4-bit phase	0.38	0.87	0.33	0.74	0.81	1.09	2.64
8-bit 95% jpeg,8-bit phase	0.32	1.00	0.32	1.36	0.76	1.75	3.87
5-bit mag^(1/2),6-bit phase	0.34	0.92	0.32	1.35	1.00	2.83	5.17
6-bit mag^(1/2),6-bit phase	0.38	0.84	0.32	1.41	0.98	2.81	5.20
7-bit mag/8, 4-bit phase	0.34	0.86	0.30	0.84	0.80	1.08	2.73
6,6-bit I/Q of FFT	0.38	0.78	0.29	1.20	0.96	10.41	12.58
8-bit 90% jpeg,8-bit phase	0.29	1.00	0.29	1.46	0.85	1.87	4.18
4-bit mag^(1/2),6-bit phase	0.31	0.91	0.28	1.14	0.97	2.93	5.04
8-bit 80% jpeg,8-bit phase	0.27	1.00	0.27	1.41	0.85	1.85	4.10
4-bit mag^(1/3),6-bit phase	0.31	0.84	0.26	1.38	1.00	5.28	7.66
8-bit mag^(1/2),4-bit phase	0.38	0.69	0.26	1.37	1.04	2.90	5.31
8-bit 95% jpeg,6-bit phase	0.26	1.00	0.26	1.52	0.86	1.91	4.29
6-bit mag^(1/2),4-bit phase	0.31	0.81	0.25	1.28	0.99	2.82	5.08
5-bit mag^(1/2),4-bit phase	0.28	0.90	0.25	1.21	1.00	2.83	5.03
4,4-bit I/Q FFT, 2 stds	0.25	0.97	0.24	0.58	0.94	10.76	12.28
3-bit Lloyd's, 5-bit phase	0.25	0.97	0.24	0.60	0.91	1.26	2.76
6-bit Lloyd's, 3-bit phase	0.28	0.86	0.24	0.98	0.95	6.55	8.48
4-bit Lloyd's, 4-bit phase	0.25	0.95	0.24	0.70	0.92	1.61	3.23
4,4-bit I/Q FFT, 1 stds	0.25	0.93	0.23	0.66	0.93	11.20	12.79
8-bit 90% jpeg,6-bit phase	0.23	1.00	0.23	1.46	0.85	1.86	4.18
5-bit Lloyd's, 3-bit phase	0.25	0.92	0.23	1.02	0.93	3.05	5.00
4-bit mag^(1/2),4-bit phase	0.25	0.89	0.22	1.01	0.97	2.93	4.91
5-bit mag^(1/2),3-bit phase	0.25	0.89	0.22	1.14	0.99	2.82	4.95
4,4-bit I/Q FFT, 3 stds	0.25	0.87	0.22	0.66	0.94	10.66	12.25
3-bit ln mag,5-bit phase	0.25	0.86	0.22	0.77	0.97	2.57	4.30
3-bit mag^(1/4),4-bit phase	0.22	0.98	0.21	0.60	0.99	5.32	6.91
3-bit ln mag,4-bit phase	0.22	0.98	0.21	0.58	0.99	2.58	4.15
4-bit mag^(1/3),3-bit phase	0.22	0.96	0.21	0.62	1.00	5.25	6.88
3-bit mag^(1/4),5-bit phase	0.25	0.83	0.21	0.81	0.98	5.30	7.09
4-bit mag^(1/3),4-bit phase	0.25	0.81	0.20	1.29	0.98	5.28	7.55
4,4-bit I/Q FFT, 4 stds	0.25	0.78	0.20	0.77	0.94	10.63	12.34
8-bit 95% jpeg,4-bit phase	0.19	1.00	0.19	1.55	0.89	1.97	4.41
8-bit mag^(1/2),2-bit phase	0.31	0.62	0.19	1.24	1.08	2.90	5.23
6-bit mag^(1/2),2-bit phase	0.25	0.77	0.19	1.10	1.00	2.83	4.94
4,4-bit I/Q of FFT	0.25	0.69	0.17	0.97	0.94	10.43	12.33
3-bit ln mag,3-bit phase	0.19	0.91	0.17	0.56	1.00	2.58	4.13
3-bit mag^(1/4),2-bit phase	0.16	0.87	0.14	0.47	1.03	5.31	6.81
2-bit mag^(1/4),1-bit phase	0.09	0.58	0.05	2.75	1.05	5.59	9.40

* JPEG in Matlab, zlib in C ** in C *** in Matlab, except Lloyd's and FFT methods in C

4.4 CCD Results

This section presents the CCD results. The first two subsections examine the data from the point of view of comparing the two CCD implementations and the four measures of CCD performance discussed in Section 1, and how each behaves in the presence of degraded imagery. The next subsection examines the data from the point of view of evaluating the usefulness of the various lossy compression schemes for images to be used in CCD. The last subsection examines the data from the second set of flights to investigate the behavior of less-correlated image pairs.

4.4.1 Comparing CCD Implementations

We ran both CCD processor implementations with 12 different lossy compression schemes and 106 image pairs from the November 1 data set. Both processors occasionally had issues where the correlation increased as the images were degraded – the Matlab code saw this occur 25 times, and the C code only twice. All of these cases have been thrown out. The data shown in the following two figures comes from the remaining 79 image pairs. Note that after these 106 image pairs were processed as a representative set from both processors, we used only the C-code CCD processor for the remainder of the images because it runs much more quickly.

Figure 5 plots the mean correlation in the Matlab output versus the mean correlation in the C code output. The different colors represent the twelve different lossy compression schemes included in this test. We see that most of the cases are roughly scattered about the line where the two results are equal. There is one set of dots far to the left that represent an image pair which was not properly registered by GA's code where the Matlab code was successful.

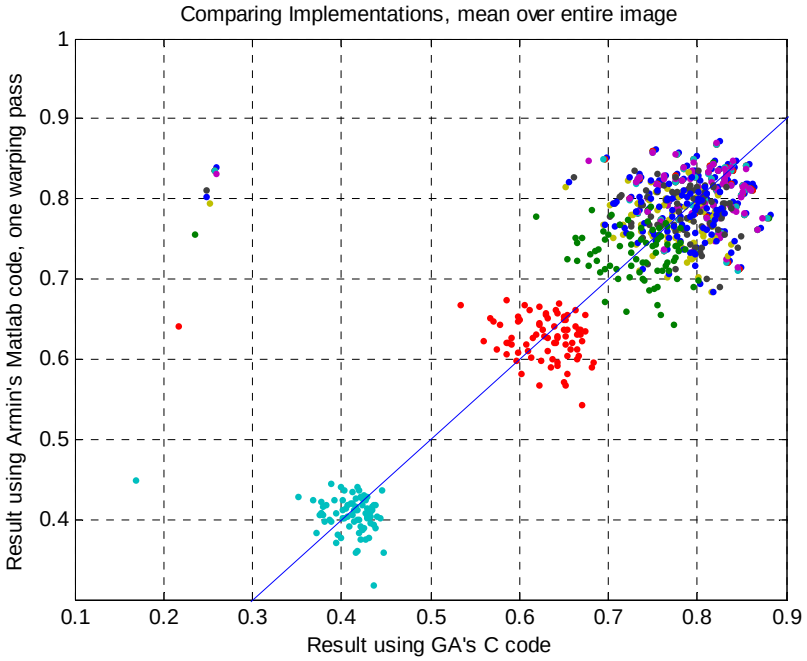

Figure 5. A comparison of the mean correlation from GA's GEARSDriver code and Armin's Matlab code.

Figure 6 shows the same data but with each measurement relative to the un-degraded version. Now, for nearly all cases, the measurements are clustered more closely about the line where the

two results are equal. For the two worst compression schemes (green and red dots), the dots that were far to the left in the previous plot now fall a good distance to the right. The worst compression scheme (red dots) also has many cases where the Matlab code is significantly worse than GA's code. However, for all cases where the CCD processing can be considered successful, the relative performance of the two implementations was indistinguishable. Thus, we postulate that the results discussed in Section 4.4.3 can be applied to other CCD processors with confidence that the performance will be similar if not identical.

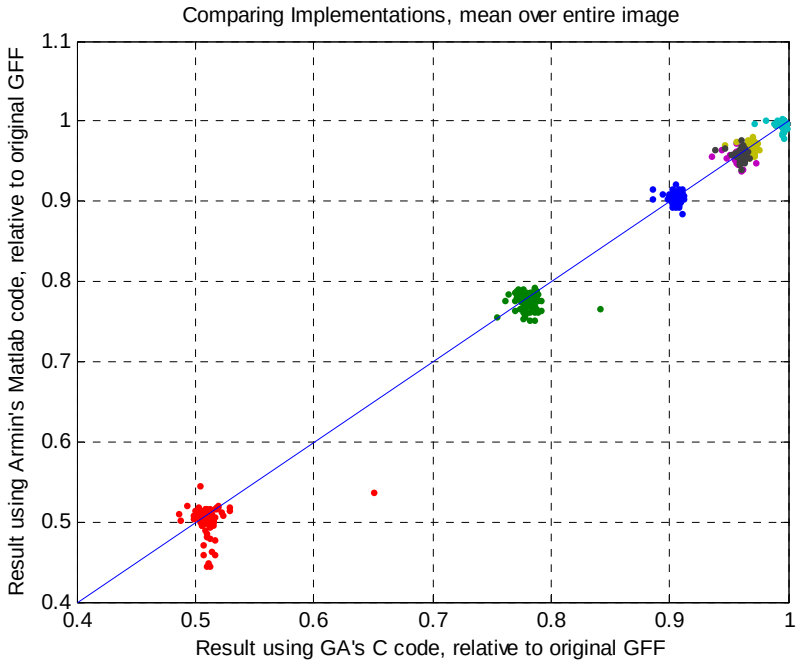

Figure 6. A comparison of the mean correlation from GA's GEARSDriver code and Armin's Matlab code, each measured relative to the original GFF image in the same processor.

4.4.2 Comparing Performance Measurements

This section compares the four performance measurements described in Section 1. In these comparisons, we show only those lossy compression schemes not indicated as "Bad" in Table 2. Each plot includes 519 image pairs. The various colors of dots indicate the different compression schemes, although the colors are repeated since we use only 7 colors for all of the compression schemes tested. The main reason to show this data is to show that the four different measures of performance are not entirely independent and that we can probably use any of them in determining which lossy compression schemes are valid for a particular application.

Figure 7 compares the total correlation averaged over only the bright regions of the image with the total correlation averaged over the entire image. As mentioned in Section 1, bright regions are defined as those 30x30-pixel regions whose minimum correlation in the original CCD product is greater than 0.7. Note that this does not refer to bright radar return but bright CCD product. As would be expected, the correlation averaged over the bright regions is greater than the average over the entire image, except for one case which may indicate bad registration. Now, if we take the same two measurements relative to the measurement for the original CCD product, then they align much more closely. This is shown in Figure 8. While there may be a

few more values above the unity line than below it, these results in general are very close to equal.

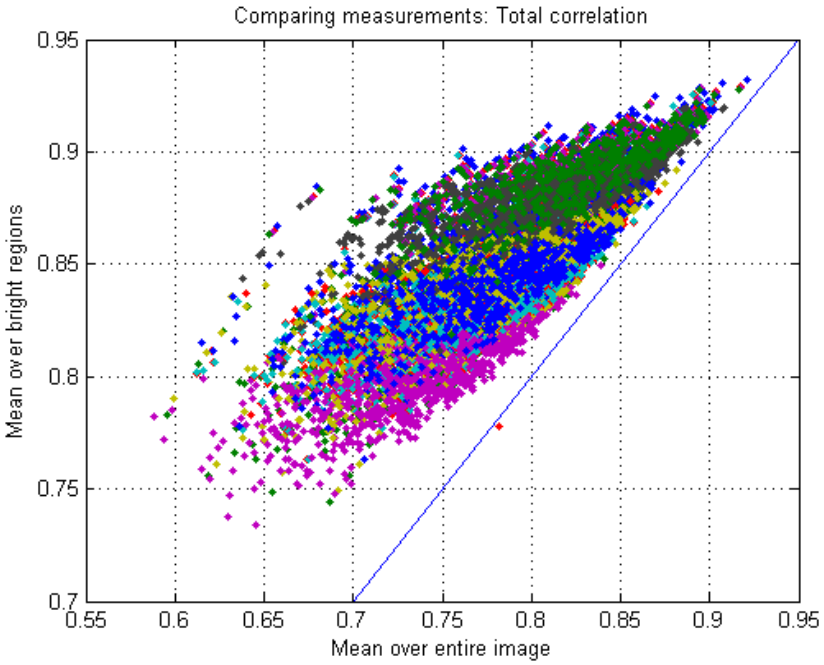

Figure 7. The total correlation, averaged over the entire image vs. averaged over only bright regions. The solid line shows where the two values are equal.

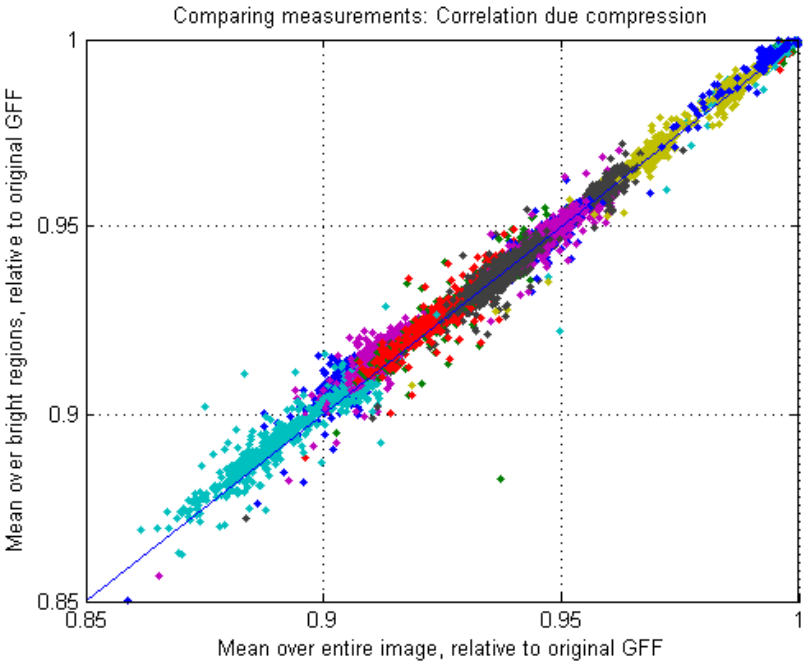

Figure 8. The correlation relative to the original GFF, averaged over the entire image vs. averaged over only bright regions. The solid line shows where the two values are equal.

Now we convert the total correlation to its SNR equivalent. Figure 9 compares the mean SNR equivalent over bright regions to the mean over the entire CCD image. Figure 10 makes a

similar comparison but with all values relative to the original un-degraded CCD product. Note that in both cases the solid line is where the mean over bright regions is 1.5 times the mean over the entire image. In Figure 9, the data fall roughly around this line, and in Figure 10 they cluster more closely around it. Figure 11 shows the difference of 1.5 times the mean over the entire image minus the mean over the bright regions. From this, we can see that there is significant variation in the difference between these two measurements. However, on average we can expect that the ratio between them is approximately 1.5.

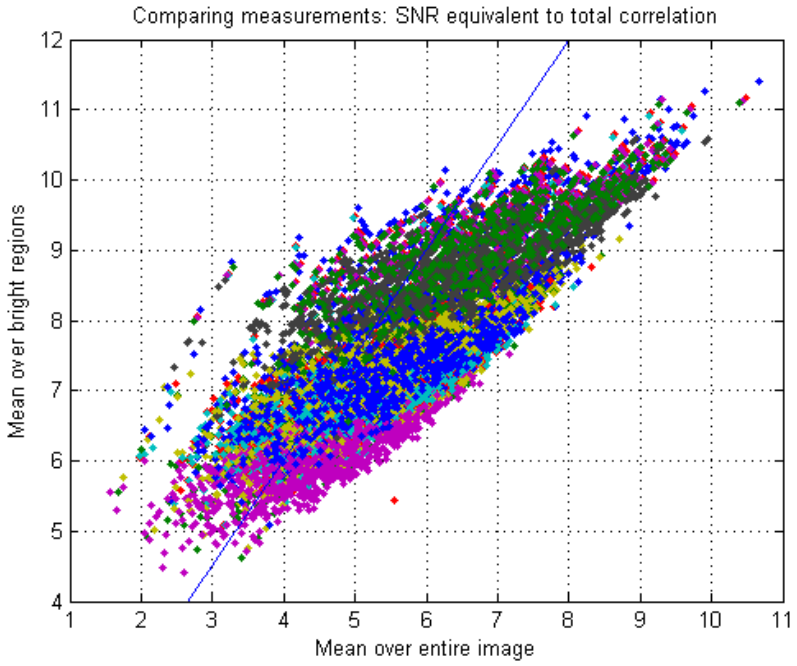

Figure 9. SNR equivalent to total correlation, averaged over the entire image vs. averaged over only bright regions. The solid line is where the mean over bright regions is 1.5 times the mean over the entire image.

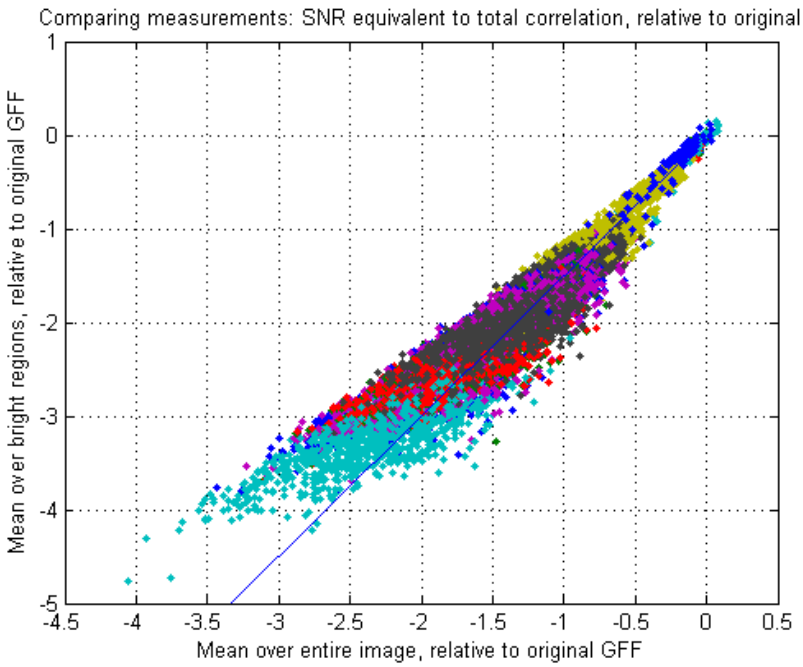

Figure 10. SNR equivalent to total correlation relative to original GFF, averaged over the entire image vs. averaged over only bright regions. The solid line is where the mean over bright regions is 1.5 times the mean over the entire image.

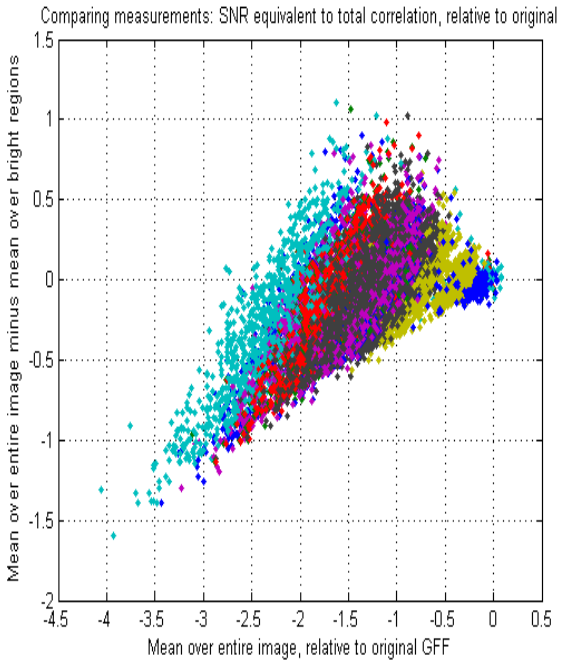

Figure 11. SNR equivalent to total correlation relative to original GFF, difference of 1.5 times the mean over entire image minus the mean over bright regions.

Figure 12 compares the loss of equivalent SNR in the bright regions to the loss of contrast between bright and dark regions. Recall that the contrast between bright and dark regions is simply measured as the difference in equivalent SNR between the two regions. Thus this plot

compares something of the form $\Delta(a-b)$ to something of the form $\Delta(a)$, where a represents the same thing in both cases. Note that the solid line is where loss of bright-dark contrast is half the loss of SNR in bright regions. Figure 13 shows the difference between the loss of equivalent SNR and two times the loss of contrast. Again, significant variation remains, but we can conclude that on average the loss in contrast is approximately half the loss in equivalent SNR in the bright regions. This is good news – since the dark regions are getting darker while the bright regions are getting darker, the ability to detect changes does not degrade as quickly as the equivalent SNR alone would suggest.

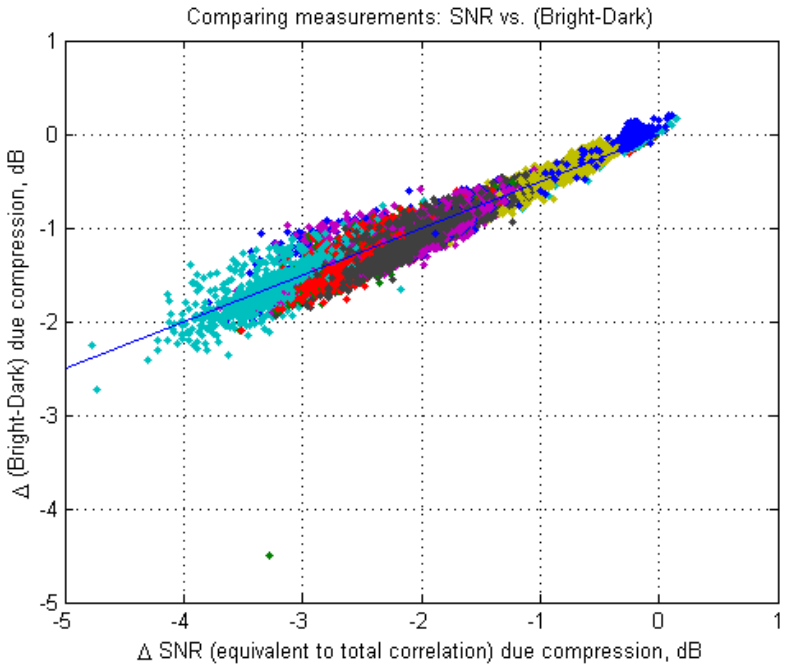

Figure 12. Loss of contrast between bright and dark regions vs. loss of SNR in the bright regions. The solid line is where loss of bright-dark contrast is half the loss of SNR in bright regions.

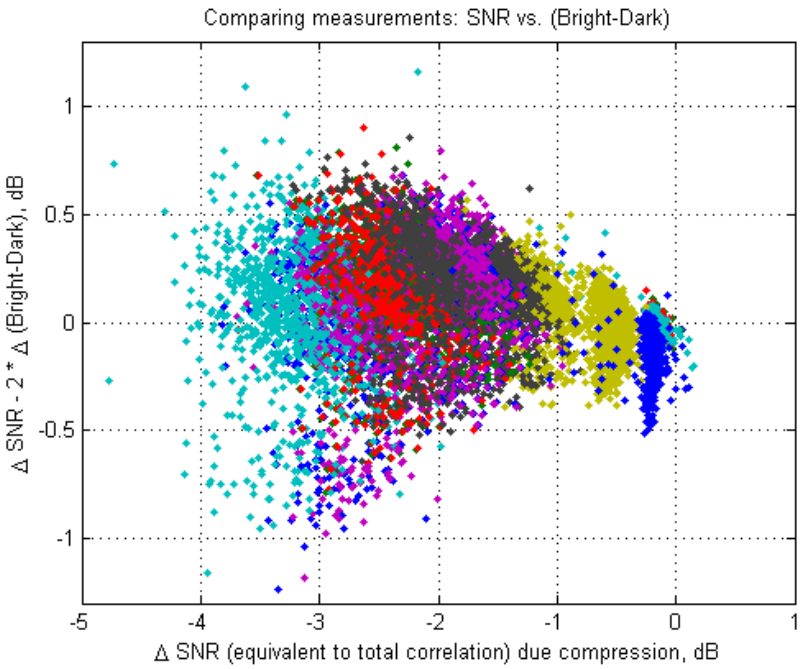

Figure 13. Loss of SNR in bright regions minus two times the loss of contrast between bright and dark regions.

Figure 14 compares the RMS of the difference between the two CCD images to the total correlation over bright regions. The solid line is where the RMS correlation difference is equal to the loss in correlation due to compression. The RMS correlation difference is well above the line for those cases with high average correlation. This seems to indicate that when we first begin degrading the images, the CCD product begins to change but its average does not. This could be similar to SAR images from slightly different viewing angles: the speckle changes but average RCS values and the bulk of the scene stay the same. This type of change was also observed in the qualitative measurements described in Section 4.1. It appears that the RMS correlation difference is lower than the loss in total correlation for the lowest-correlation cases to the left of the plot, although we don't go out far enough to be certain that this is the trend. Figure 15 shows the difference between the RMS of the correlation difference and the loss in correlation over the bright regions. As in the other comparisons above, there is some variation in the data, but on average the two values are fairly close, especially for cases with significant degradation.

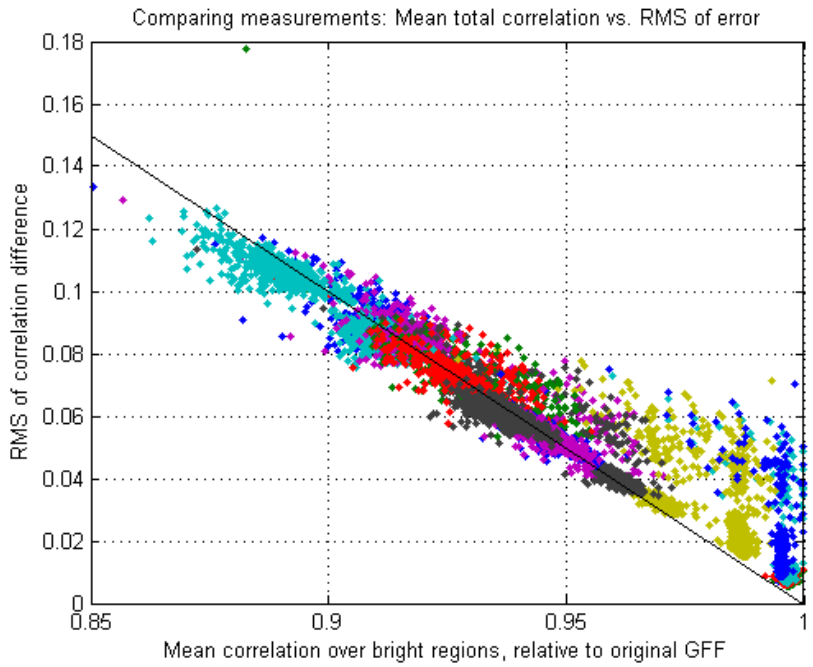

Figure 14. RMS of correlation change over entire image vs. mean correlation over bright regions, relative to original GFF. The solid line is where the RMS of correlation change is equal to the loss in mean correlation.

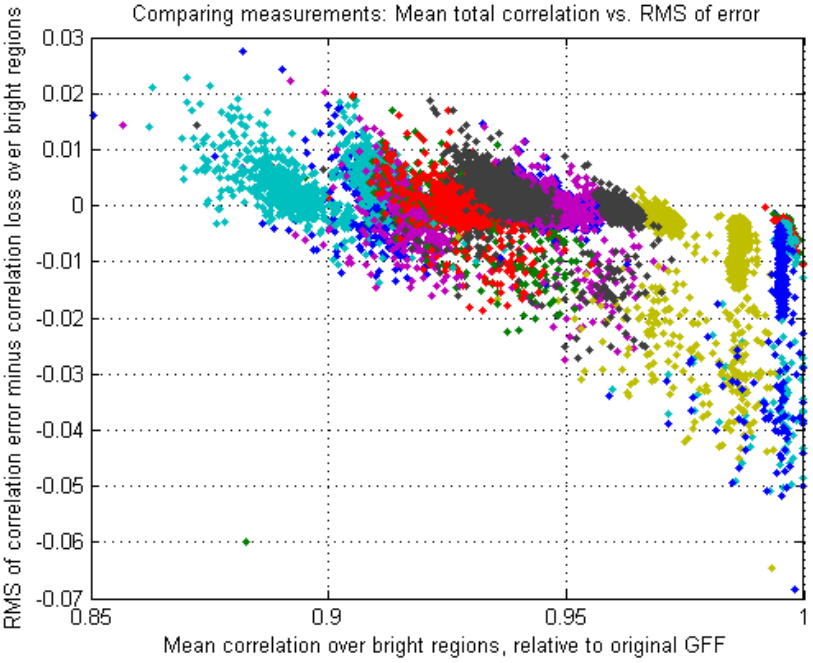

Figure 15. RMS of correlation error minus mean loss of correlation over bright regions.

4.4.3 Evaluating Lossy Compression Methods

This section discusses the relative performance of the various lossy compression methods examined in this study. Table 6 shows the mean values for the four performance measures examined, as well as the compression ratios. The chart is sorted in order of decreasing total compression ratio, assuming that the data is packed and compressed as discussed in Section 4.2. Systems choosing not to use zlib or similar compression should use the first column as the compression ratio instead of the third. The colors in the chart indicate those cases which meet the thresholds indicated in Table 5. Note that these colors loosely follow a green-yellow-red progression from good to bad. All the thresholds except those indicated by red entries require the measurements to be smaller than the indicated values. The green entries indicate a threshold which is in some sense subjective and will be discussed below.

Table 5. The performance thresholds and the corresponding colors as shown in Table 6.

	Thresholds		
Color	Δ SNR, entire image	Δ SNR, bright regions	Δ (Bright-Dark)
Green	no visible change, << yellow thresholds, single-mode histogram		
Tan	0.667	1.0	0.5
Yellow	1.0	1.5	0.75
Orange	1.33	2.0	1.0
Red	>3.33	>5.0	>2.5

Table 6. CCD results given lossy compression. Colors indicate performance thresholds given in Table 5. "*" indicates cases averaged over the entire image, while "" indicates cases averaged over 30x30 blocks with minimum correlation > 0.7.**

	Lossy Compression ratio	Zlib compression ratio	Total compression ratio	Mean correlation due compression*	Delta SNR due compression, dB*	Mean correlation due compression**	Delta SNR due compression, dB**	Delta (bright-dark) due compression, dB	RMS of difference in correlation
16-bit mag,16-bit phase	1	0.53	0.53	1	0	1	0	0	0
8-bit mag^(1/2),12-bit phase	0.63	0.68	0.43	1	-0.09	1	-0.13	-0.07	0.007
8-bit mag^(1/2),8-bit phase	0.5	0.78	0.39	1	-0.09	1	-0.14	-0.07	0.007
9-bit mag/2, 4-bit phase	0.41	0.93	0.38	0.99	-0.26	0.99	-0.41	-0.24	0.019
8-bit mag/2, 4-bit phase	0.38	0.92	0.34	0.99	-0.27	0.99	-0.41	-0.22	0.02
8-bit mag^(1/2),6-bit phase	0.44	0.75	0.33	1	-0.1	1	-0.16	-0.09	0.007
8-bit mag/4, 4-bit phase	0.38	0.87	0.33	0.99	-0.27	0.99	-0.42	-0.24	0.019
8-bit 95% jpeg,8-bit phase	0.3	1	0.32	0.95	-1.25	0.95	-1.84	-1.01	0.052
5-bit mag^(1/2),6-bit phase	0.34	0.92	0.32	1	-0.12	1	-0.18	-0.09	0.014
6-bit mag^(1/2),6-bit phase	0.38	0.84	0.32	1	-0.12	1	-0.18	-0.1	0.012
7-bit mag/8, 4-bit phase	0.34	0.86	0.3	0.99	-0.31	0.99	-0.46	-0.25	0.02
6,6-bit I/Q of FFT	0.38	0.78	0.29	0.99	-0.18	0.99	-0.24	-0.09	0.014
8-bit 90% jpeg,8-bit phase	0.28	1	0.29	0.92	-1.72	0.93	-2.45	-1.29	0.073
4-bit mag^(1/2),6-bit phase	0.31	0.91	0.28	0.99	-0.17	0.99	-0.22	-0.04	0.021
8-bit 80% jpeg,8-bit phase	0.27	1	0.27	0.89	-2.45	0.89	-3.38	-1.67	0.107
4-bit mag^(1/3),6-bit phase	0.31	0.84	0.26	0.97	-0.77	0.97	-1.13	-0.61	0.034
8-bit mag^(1/2),4-bit phase	0.38	0.69	0.26	0.99	-0.33	0.99	-0.52	-0.31	0.016
8-bit 95% jpeg,6-bit phase	0.24	1	0.26	0.95	-1.26	0.95	-1.85	-1.02	0.053
6-bit mag^(1/2),4-bit phase	0.31	0.81	0.25	0.99	-0.35	0.99	-0.54	-0.32	0.02
5-bit mag^(1/2),4-bit phase	0.28	0.9	0.25	0.99	-0.35	0.99	-0.54	-0.3	0.021
4,4-bit I/Q FFT, 2 stds	0.25	0.97	0.24	0.94	-1.39	0.95	-1.84	-0.74	0.072
3-bit Lloyd's, 5-bit phase	0.25	0.97	0.24	0.98	-0.55	0.98	-0.79	-0.38	0.029
6-bit Lloyd's, 3-bit phase	0.28	0.86	0.24	0.96	-0.99	0.96	-1.49	-0.86	0.042
4-bit Lloyd's, 4-bit phase	0.25	0.95	0.24	0.98	-0.41	0.98	-0.63	-0.35	0.022
4,4-bit I/Q FFT, 1 stds	0.25	0.93	0.23	0.79	-4.11	0.8	-5.27	-2.07	0.207
8-bit 90% jpeg,6-bit phase	0.22	1	0.23	0.92	-1.73	0.93	-2.46	-1.3	0.074
5-bit Lloyd's, 3-bit phase	0.25	0.92	0.23	0.96	-1.02	0.96	-1.54	-0.88	0.043
4-bit mag^(1/2),4-bit phase	0.25	0.89	0.22	0.98	-0.4	0.99	-0.57	-0.25	0.026
5-bit mag^(1/2),3-bit phase	0.25	0.89	0.22	0.96	-1.02	0.96	-1.54	-0.87	0.045
4,4-bit I/Q FFT, 3 stds	0.25	0.87	0.22	0.97	-0.76	0.97	-1.01	-0.4	0.042
3-bit ln mag,5-bit phase	0.25	0.86	0.22	0.94	-1.34	0.94	-1.94	-1.05	0.059
3-bit mag^(1/4),4-bit phase	0.22	0.98	0.21	0.91	-2.09	0.91	-2.88	-1.41	0.09
3-bit ln mag,4-bit phase	0.22	0.98	0.21	0.94	-1.49	0.94	-2.14	-1.15	0.064
4-bit mag^(1/3),3-bit phase	0.22	0.96	0.21	0.93	-1.57	0.93	-2.28	-1.23	0.065
3-bit mag^(1/4),5-bit phase	0.25	0.83	0.21	0.91	-1.96	0.92	-2.7	-1.32	0.085
4-bit mag^(1/3),4-bit phase	0.25	0.81	0.2	0.96	-0.97	0.96	-1.43	-0.78	0.042
4,4-bit I/Q FFT, 4 stds	0.25	0.78	0.2	0.96	-1.05	0.96	-1.4	-0.55	0.055
8-bit 95% jpeg,4-bit phase	0.18	1	0.19	0.94	-1.45	0.94	-2.11	-1.15	0.059
8-bit mag^(1/2),2-bit phase	0.31	0.62	0.19	0.85	-3.13	0.85	-4.33	-2.19	0.14
6-bit mag^(1/2),2-bit phase	0.25	0.77	0.19	0.85	-3.14	0.85	-4.34	-2.2	0.141
4,4-bit I/Q of FFT	0.25	0.69	0.17	0.91	-2.04	0.92	-2.68	-1.05	0.101
3-bit ln mag,3-bit phase	0.19	0.91	0.17	0.91	-2.04	0.91	-2.89	-1.52	0.086
3-bit mag^(1/4),2-bit phase	0.16	0.87	0.14	0.78	-4.28	0.78	-5.64	-2.54	0.197
2-bit mag^(1/4),1-bit phase	0.09	0.58	0.05	0.51	-8.24	0.51	-10.08	-3.56	0.467

Figure 16 shows the histograms of the RMS of the correlation difference for all cases that meet the green, tan, and yellow thresholds in all three columns. The color of the subplot title indicates the threshold color code from the first colored column. In many cases, there are two different behaviors. To distinguish the two, we find the "histogram breakpoint" as the first bin without any occurrences, lying to the right of the primary hump. The blue portion of the histogram shows occurrences to the left of this point, and the red portion shows occurrences to the right. The text within each histogram shows the location of the breakpoint and the mean value for each portion of the histogram. Examination of a few cases indicates that the red portion of each histogram corresponds to cases where the registration is incorrect due to the image degradation, and the blue portion corresponds to cases where the registration is maintained. The histograms have been cut off at the top to allow viewing the detail in the tails.

Notice that the cases coded green have all the values clumped together with a small RMS correlation error. The 8-bit square-root of magnitude, 4-bit phase case also has all the data grouped together with no registration errors, but the error is a little higher. We conclude that 8 bits of square-root of magnitude data and at least 4 bits of phase are required for registration as reliable and accurate as with 16-bit magnitude, 16-bit phase data. With only 4, 5, or 6 bits of square-root of magnitude data, or with other operations, we sometimes obtain good registration results, but they are not as reliable. In these histograms, we can see the non-subjective part of the green color-coded cases: the histograms of performance of these cases are clustered tighter and are significantly lower than any other cases. For this reason, we recommend 8-bit square-root of magnitude and 6-bit phase data for projects which desire low-data-rate images with no CCD degradation. For projects which can allow a small amount of CCD degradation, we recommend 8-bit square root of magnitude and 4-bit phase data for a slightly smaller data rate. For projects which desire 8 total bits per pixel at the expense of greater CCD degradation, we recommend 4-bit magnitude data with Lloyd's quantizer and 4-bit phase data. This is also the recommended choice for projects which can accommodate the loss in CCD performance but have processing throughput limitations or other reasons to avoid using lossless compression. Note that the total compression for 8-bit square-root of magnitude/4-bit phase data is almost as good as the 4-bit Lloyd, 4-bit phase data when we include the effects of the lossless compression, but when we include only the lossy compression ratio the 4-bit/4-bit case is significantly better.

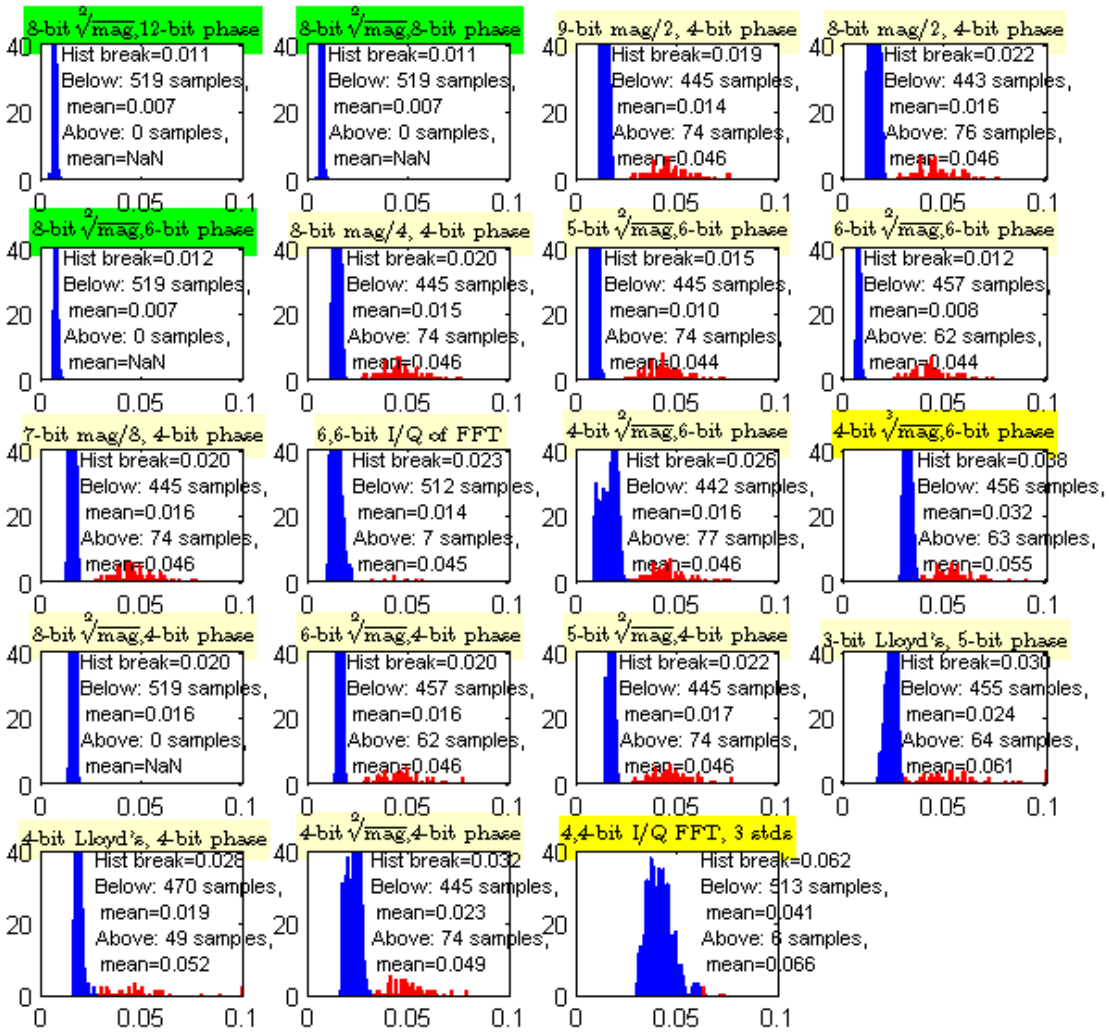

Figure 16. Histograms of the cases marked green, tan, and yellow in Table 6.

4.4.4 CCD results versus original correlation

All of the CCD data analyzed in Section 4.4.3 was from a single flight with two scene locations. We desire to compare these results to those at other locations. The other locations we had available were from a series of three flights by a different radar. From these flights, we processed 75 pairs, of which 4 had increasing correlation as the images degraded and 12 had generally poor registration. In this section, we consider the remaining 59 image pairs. Figure 17 shows a histogram of the un-degraded correlation of all 59 pairs. There is a much wider spread of values than was seen in the previous data set. We examined the motion records from each pair and found no evidence of differential motion that would degrade the correlation significantly. The greatest difference in depression angles occurs in the cases colored blue in the histogram – those with the best correlation. The common difference between the cases colored blue and those colored green and red is the amount of time between the reference pass and the comparison pass. There is one case with a correlation of 0.59 that we might think should be included in the green set; however, its other characteristics (including time between

passes) align more closely with the blue cases. We therefore consider it an anomalous member of the blue set. We will see below that the blue and green cases behave rather similarly, so it is not terribly significant which set a particular case is grouped with.

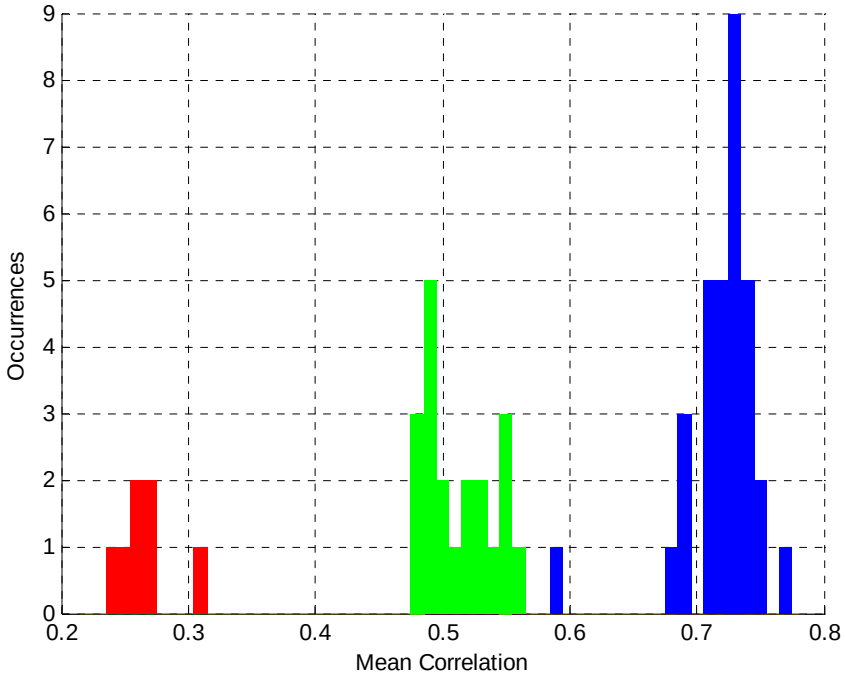

Figure 17. A histogram of correlation from 59 image pairs at several sites. Colors indicate different groups of pairs discussed in the text.

All the CCD pairs shown in blue in Figure 17 come from the same flight. Those colored red and green come from flights 4 days apart. During the four days between these flights, there was significant rain in the area, so the lower correlation is probably due to actual detected change. The cases colored green and blue are from Sandia's radar calibration range, while the red cases are from farmland and other areas some distance away near the Rio Grande. We believe that the extreme low correlation observed in the red cases is due to an abundance of trees and their shadows in these areas, as well as ground cover that is more susceptible to change due to rain (e.g. farmland vs. bare desert).

Figure 18 compares the average correlation due to the lossy compression for the three groups of CCD pairs in Figure 17 and the data presented in the previous section. In adjusting the axis to make some of the details more easily visible, we have cut off some of the lowest-correlation data. Note that especially for the cases with correlation above about 0.94, the CCD pairs colored blue and green in Figure 17 very closely approximate the data in the previous section. Those cases colored red in Figure 17 tend to be slightly higher. We believe that this is because the red cases already have such low correlation that they are not affected as much by further degradations in the images.

We believe it is significant that the change in correlation given the lossy compression is so nearly the same for cases with average correlation ranging all the way from 0.48 up to 0.92. Many if not all of the cases presented in the previous section have lower correlation

than would be expected given the clutter-to-noise ratio estimated in the images. Since images with initial correlations over such a wide range have shown similar behaviors given these lossy compression schemes, we expect that image pairs producing a yet higher correlation will most likely behave in the same manner.

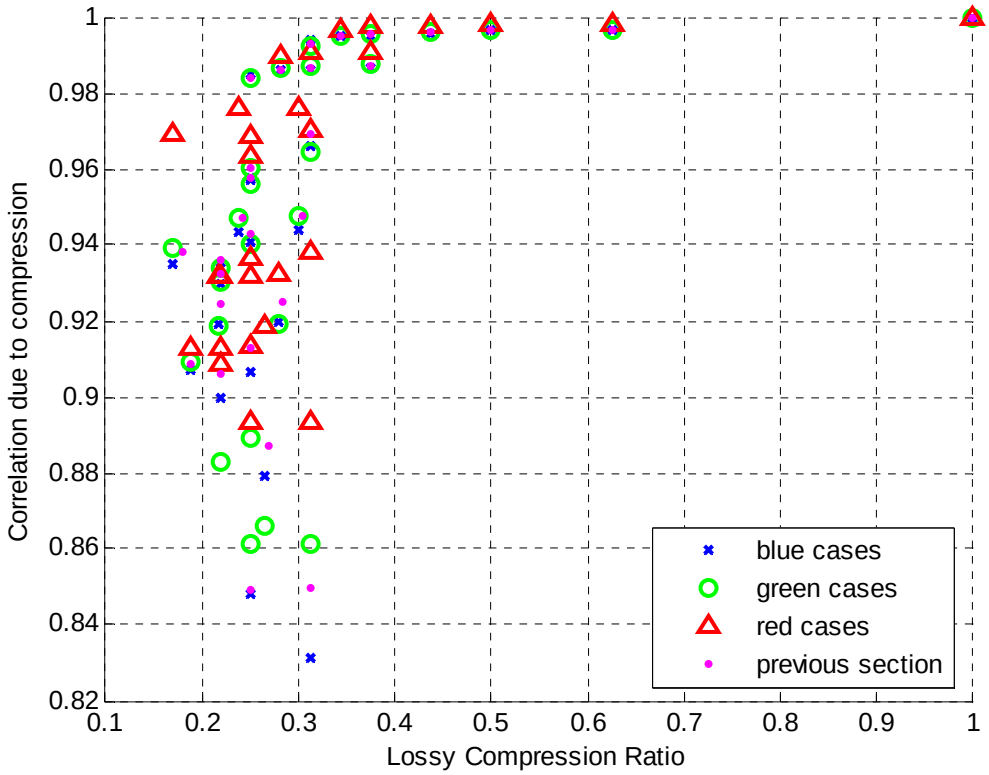

Figure 18. A comparison of the three groups of CCD pairs in this section with the cases in the previous section.

5 A few Theoretical Notes

This section covers several theoretical issues related to predicting the correlation in the presence of lossy compression schemes.

5.1 The Effect of Window Size in Correlation Estimation

This section briefly reviews the calculation of correlation in CCD and how the window size affects the estimate. The comments in this section are well known in the CCD world, but we include them here as a reminder.

Before we calculate the correlation, the two images must be co-registered so that one lies exactly on top of the other, to precision well below a pixel. We then calculate the correlation as

$$\rho = \frac{\left|\sum s_1 s_2^*\right|}{\sqrt{\sum s_1 s_1^* \cdot \sum s_2 s_2^*}} \tag{6}$$

where s_1 and s_2 represent the two complex, registered images. The sums are taken over some window size in the image, typically using an odd number of pixels and the same number of pixels in each dimension. 5x5 and 7x7 are common values for the window size.

Once we degrade CCD images through lossy compression as discussed in this document, we may desire to ask, "Why don't we increase the window size to regain correlation at the expense of resolution in the CCD product?" However, we must recall that Equation (6) is really averaging the correlation over a window size, not averaging to increase SNR. Averages only increase SNR when the noise is a zero-mean random variable that is added to the signal. Due to the multiplications involved in Equation (6), the undesired quantities never appear in this form. Increasing the window size only decreases the variance of the estimate of the correlation, not affecting the mean at all.

Confusion may arise due to a similar use of correlation in IFSAR. IFSAR height mapping uses an equation similar to Equation (6), but takes the phase instead of the magnitude of the result. In that case, increasing the window size results in lower height noise at a cost of worse resolution (or increased post spacing) in the resulting product. This is because the height noise is related to the variance in the correlation estimate. As stated above, we can affect the variance but not the mean of the correlation estimate by changing the window size.

Assuming we are already using the proper algorithms, there is no way to increase the average correlation in a particular image pair to compensate for lossy compression. Another way we might consider attempting this is to average in the image domain to bring up the SNR before doing the correlation. This would work if the image were correlated over a large enough region to perform the average (and if the noise remained uncorrelated over this region). Typical Sandia radars produce images with pixels 1.2

times smaller than the resolution cell. This means that the images are only correlated to a distance of 1.2 pixels. We could adjust the parameters during the collection so that the resulting images were more correlated, but this results in a larger number of pixels for a given resolution, negating the size gains we would have gotten from lossy compression. We also might think we can increase the correlation of a given image through reduced resolution sub-band processing or a similar method. This might work if we could separate the noise from the signal during this process, but since we cannot do that (if we could, we would just keep the noise separate and have a perfect image anyway) we end up increasing the correlation of the noise in the same proportions as we do for the signal, with no net effect.

5.2 The Effect of Phase Quantization

Assume we have two GFF images of the same area, taken at different times (i.e. a CCD pair). The ideal complex image for both cases is denoted *s*. Due to many factors, the images created are different from the ideal (and different from each other). These factors may include geometry differences, scene changes, focusing differences, and others. We denote the difference from the ideal as n_1 for the first image and n_2 for the second image, and the actual images as $s_1 = s + n_1$ and $s_2 = s + n_2$. Now assume that we quantize the phase for both images to N_p bits. Since the phase is a uniform random variable between -pi and +pi, the quantization noise introduced is a uniform random variable. The phase is quantized as

$$p_{quantized} = 2\pi \cdot \frac{\text{round}\left(\frac{p_{GFF} \cdot 2^{N_p}}{2^{16}} \right)}{2^{N_p}} \tag{7}$$

compared to the original phase,

$$p_{original} = 2\pi \cdot \left(\frac{p_{GFF}}{2^{16}} \right) \tag{8}$$

where p_{GFF} is the 16-bit representation of the phase in the GFF file. With a uniform distribution as the input, the *round* function produces an error uniformly distributed between -0.5 and +0.5. Thus, the phase error is

$$p_{error} = 2\pi \cdot \frac{U\left(-0.5 \ldots + 0.5\right)}{2^{N_p}} = U\left(-\frac{\pi}{2^{N_p}} \ldots \frac{\pi}{2^{N_p}} \right) \tag{9}$$

where U( ) represents a uniform distribution between the indicated values. We denote the phase errors for the two images as p_1 and p_2.

Now we desire to calculate the cross-correlation between the two images in CCD processing. The equation used to calculate this is

$$\gamma = \frac{\left|\sum s_1 s_2^*\right|}{\sqrt{\sum s_1 s_1^* \cdot \sum s_2 s_2^*}} \tag{10}$$

where the sums are taken over some region of the image. The intent of these sums is to average out noise in the estimate of correlation. For this derivation, we assume that the summation symbol indicates a statistical expectation, forgetting that we really are summing over a small (say, 5x5 or 7x7) region of the image. The correlation for our case is then

$$\gamma = \frac{\left|\sum \left(s+n_1\right) e^{jp_1} \left(s+n_2\right)^* e^{-jp_2}\right|}{\sqrt{\sum \left|s+n_1\right|^2 \left|e^{jp_1}\right|^2 \cdot \sum \left|s+n_2\right|^2 \left|e^{jp_2}\right|^2}} = \frac{\left|\sum \left(\left|s\right|^2 + s^* \cdot n_1 + s \cdot n_2^* + n_1 \cdot n_2^*\right) e^{j(p_1-p_2)}\right|}{\sqrt{\sum \left|s+n_1\right|^2 \cdot \sum \left|s+n_2\right|^2}} \tag{11}$$

Since *s*, n_1, and n_2 are mutually uncorrelated, all the cross terms in the numerator go away in the expectation. The magnitude-squares in the denominator have similar expansions and similar cross-terms which disappear in the expectation. We also simplify by assuming that the two noise terms have identical statistics (i.e. the effective SNR of both images is the same). Thus, the correlation simplifies to

$$\gamma = \frac{\left|\sum \left(\left|s\right|^2 \cdot e^{j(p_1-p_2)}\right)\right|}{\sum \left|s\right|^2 + \sum \left|n_1\right|^2} \tag{12}$$

The denominator in this form is the same as for the original non-quantized correlation, so we focus for the moment on the numerator. Notice that we are taking the complex magnitude of a sum. In the ideal case, each term of the sum is real. However, we have added a phase term to each one. From a statistical point of view, we may want to say that the sum vanishes because p_1 and p_2 (and their difference) are uncorrelated with s. However, since p_1 and p_2 are small, $e^{j(p_1-p_2)}$ has a mean of 1.0 and a complex variation about this value. With two non-zero-mean uncorrelated quantities, the expectation is the product of the two individual expectations, or

$$\gamma = \frac{\sum \left|s\right|^2 \cdot \left|\sum e^{j(p_1-p_2)}\right|}{\sum \left|s\right|^2 + \sum \left|n_1\right|^2} \tag{13}$$

Now we focus on the term $\left|\sum e^{j(p_1-p_2)}\right|$. Since p_1 and p_2 are small, zero-mean, and uniformly distributed, their difference has a triangle-function probability distribution with a peak at zero. If we picture the complex plane, each term in the sum is a vector of length one and direction nearly parallel to the real axis. On average, the result of the entire sum is real, so only the real part of each term contributes. The real part is $\cos\left(p_1 - p_2\right)$. Since the argument of the cosine is always small, we expand this in its Taylor series so that the entire term becomes

$$\left|\sum e^{j(p_1-p_2)}\right| = \sum\left(1-\frac{(p_1-p_2)^2}{2}\right) = 1-\frac{\sum(p_1-p_2)^2}{2} = 1-\frac{\sum p_1^2+\sum p_2^2}{2} \tag{14}$$

Notice that we have assumed that p_1 and p_2 are uncorrelated. Recalling that the two phase-error terms are identically distributed, the correlation now simplifies to

$$\gamma = \frac{\sum|s|^2\left(1-\sum p_1^2\right)}{\sum|s|^2+\sum|n_1|^2} \tag{15}$$

We can now separate the correlation due to phase quantization from the original correlation as

$$\gamma_{quantizedPhase} = \gamma_{original}\gamma_{phaseQuantization} \tag{16}$$

where

$$\gamma_{phaseQuantization} = 1-\sum p_1^2 \tag{17}$$

This is simply one minus the variance in this zero-mean, uniformly distributed random variable. Since we know the distribution from Equation (9), we can easily calculate the variance as

$$\sum p_1^2 = \frac{1}{12}\cdot\left(\frac{2\pi}{2^{N_p}}\right)^2 = \frac{1}{3}\cdot\left(\frac{\pi}{2^{N_p}}\right)^2 \tag{18}$$

so that

$$\gamma_{phaseQuantization} = 1-\frac{1}{3}\cdot\left(\frac{\pi}{2^{N_p}}\right)^2 \tag{19}$$

5.2.1 Comparing Theory to Reality

In this section, we compare the theoretical result derived above to experimental results. First, we created a set of synthetic images, following the models described above. The original image *s* has Gaussian real and imaginary parts, as do the noise terms n_1 and n_2. These three terms are scaled to create a desired effective SNR, and we create the two synthetic images. In order to more fully model the differences between real CCD images, we multiply the second image by a random phase and by a random magnitude between approximately 0.89 and 1.12 (±0.5 dB). For each effective SNR, we create 32 such image pairs and test the correlation between them in the presence of phase quantization. The result is shown in Figure 19 (correlation) and Figure 20 (effective SNR). For all cases, Figure 19 shows the correlation for a particular test normalized by the correlation for the corresponding original undegraded image pair. Figure 20 shows this same value converted to log-space as effective SNR to separate the highly correlated cases. The

squares indicate the theoretical value (not a function of original SNR or original correlation), while the lines indicate the average value obtained from the 32 synthetic image pairs. The theoretical data matches the data from synthetic image pairs rather well, although it is slightly pessimistic for the 2- and 3-bit cases.

In addition to testing synthetic data, we performed similar tests using the actual CCD image pairs used for the data presented in Section 1. We performed a separate set of tests using 16-bit magnitude and N-bit phase, where N ranges from 3 to 16. The results of these tests are presented as the stars in Figure 19 and Figure 20. These tests have consistently higher correlation than the theoretical value or the synthetic data, which probably indicates that there is more correlation between the two phase errors in the real data than in the synthetic data. We conclude that the theoretical expression in Equation (19) is a good estimate but slightly pessimistic compared to real images.

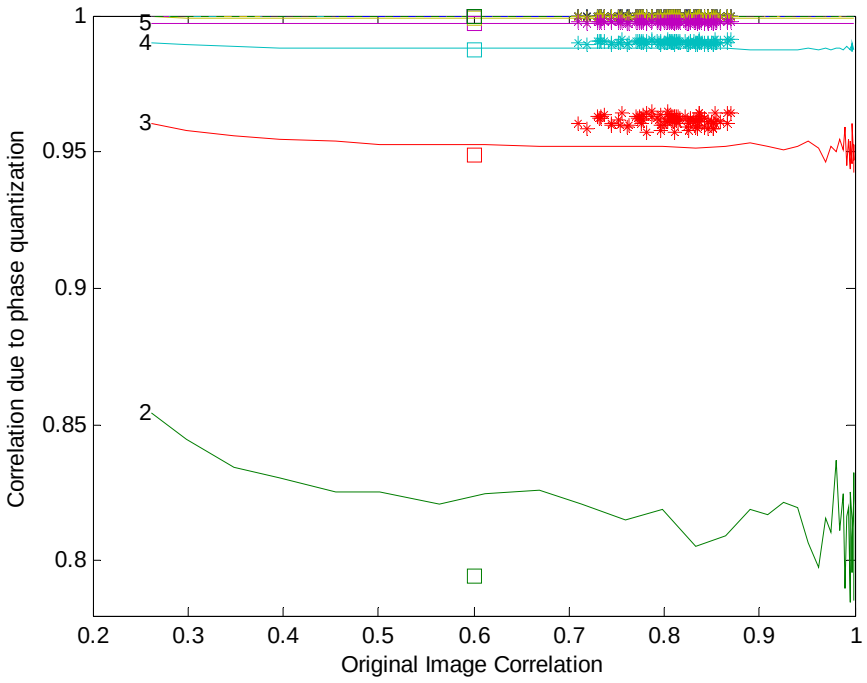

Figure 19. Theoretical correlation due to phase quantization (squares) compared to synthetic images (lines) and actual CCD data (stars). Numbers to the left of each line indicate number of phase bits stored.

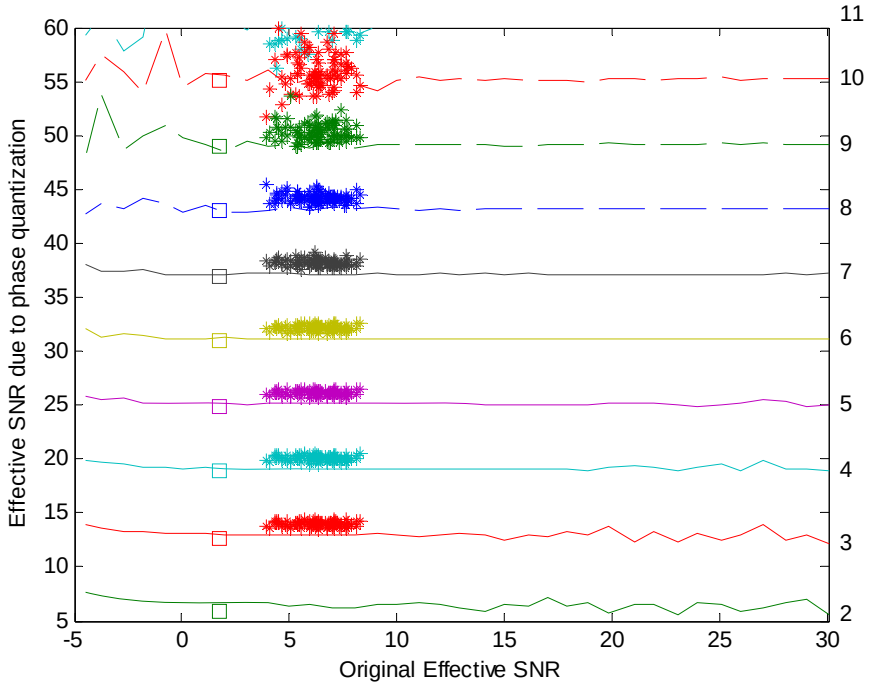

Figure 20. Theoretical effective SNR due to phase quantization (squares) compared to synthetic images (lines) and actual CCD data (stars). The numbers at the right indicate the number of phase bits stored.

5.3 The Effect of Magnitude Quantization

As above, assume we have two GFF images of the same area, taken at different times (i.e. a CCD pair). The ideal complex image for both cases is denoted *s*. Due to many factors, the images created are different from the ideal (and different from each other). These factors may include geometry differences, scene changes, focusing differences, and others. We denote the difference from the ideal as n_1 for the first image and n_2 for the second image, and the actual images as $s_1 = s + n_1$ and $s_2 = s + n_2$. Now assume that we quantize the magnitude of each image in the following manner. We take the N_r-th root of each image, divide by a scale factor k_{scale}, then quantize this value to the nearest integer. We also clip the highest-magnitude pixels by limiting them to 2^{Nm}-1 where N_m is the number of bits to store for the magnitude. We denote the data stored and transmitted for each image as

$$u_i = \left(\frac{\sqrt[N_r]{|s_i|}}{k_{scale}} + n_{qi}^{'} - n_{ci}^{'} \right) \tag{20}$$

where the "i"s are replaced by 1 and 2 for each image and the variables are

$n_{qi}^{'}$ = original quantization error -0.5...+0.5

$n_{ci}^{'}$ = original clipping error

We show a negative sign on the clipping error to emphasize that it always makes the result smaller than the original. We return to the image domain by multiplying by k_{scale}, by undoing the N_r-th root operation, and by multiplying by the phase of the original image. Thus the magnitude of the degraded image is

$$t_i = \left(k_{scale} \cdot \left(\frac{\sqrt[N_r]{|s_i|}}{k_{scale}} + n_{qi}^{'} - n_{ci}^{'} \right) \right)^{N_r} \tag{21}$$

Note that we can apply the clipping error after returning to the magnitude domain with no effect on the image quality degradation and with significantly simpler equations. Thus, we remove $n_{ci}^{'}$ from the equation and replace it with n_{ci} outside of the parentheses, so that the image magnitude simplifies to

$$t_i = \left(\sqrt[N_r]{|s_i|} + k_{scale} \cdot n_{qi}^{'} \right)^{N_r} - n_{ci} = \left(\sqrt[N_r]{|s_i|} + n_{qi} \right)^{N_r} - n_{ci} \tag{22}$$

where

n_{qi} = effective quantization error $-0.5k_{scale} \ldots + 0.5k_{scale}$

n_{ci} = effective clipping error

Note that the clipping error n_{ci} is always positive or zero and thus has non-zero mean, while the quantization error is zero mean. The magnitude-quantized complex image is then

$$im_i = \left(\left(\sqrt[N_r]{|s_i|} + n_{qi} \right)^{N_r} - n_{ci} \right) \cdot e^{j\phi_i} = \left(\sqrt[N_r]{|s_i|} + n_{qi} \right)^{N_r} \cdot e^{j\phi_i} - n_{ci} \cdot e^{j\phi_i} \tag{23}$$

Where the variables are defined as

$$\begin{aligned} im_i &= \text{image-doman representation of the actual image as stored and transmitted} \\ \phi_i &= \text{phase of the original (noisy but undegraded) image} \end{aligned} \tag{24}$$

We desire to calculate the cross-correlation between the two images in CCD processing. The equation used to calculate this is

$$\gamma = \frac{\left| \sum im_1 im_2^* \right|}{\sqrt{\sum im_1 im_1^* \cdot \sum im_2 im_2^*}} \tag{25}$$

where the sums are taken over some region of the image. The intent of these sums is to average out noise in the estimate of correlation. For these derivations, we assume that the summation symbol indicates a statistical expectation, forgetting that we really are summing over a small (say, 5x5 or 7x7) region of the image. We here encounter two

difficulties. First, it is difficult to simplify this equation for a general N_r. We thus leave the simplification to the subsections below, where we consider one value of N_r in each subsection. Second, the clipping noise term n_{ci} is difficult to handle because it is highly correlated with the original signal s_i (in fact, it is a function of s_i). In many cases, it is also of the same magnitude as s_i. Due to these complications, for the present we assume that n_{ci} is zero. In Section 5.4, we return to this term and examine its effects and the difficulty in predicting the correlation in the presence of clipping.

In each of the following subsections, we calculate the power in the quantization noise that is added to the image during the compression process, N_{qi}. We convert this to correlation by recalling an equation that can be derived from Equation (25),

$$\gamma = \frac{\mathrm{E}\left(s^2\right)}{\sqrt{\mathrm{E}\left(s^2\right)+\mathrm{E}\left(n_1^2\right)}\cdot\sqrt{\mathrm{E}\left(s^2\right)+\mathrm{E}\left(n_2^2\right)}} \tag{26}$$

In comparisons with existing image data, we calculate the signal and noise power of the image as

$$\begin{aligned}\mathrm{E}\left(s^2\right)_i &= \frac{\mathrm{E}\left(s_i^2\right)}{1+\dfrac{1}{SNR}} \\ \mathrm{E}\left(n_i^2\right) &= \frac{\mathrm{E}\left(s_i^2\right)}{1+SNR}\end{aligned} \tag{27}$$

where *SNR* is the signal-to-noise ratio, determined from the undegraded correlation. Note that the estimate of the signal power obtained from the two images may be different; thus, the *i* subscript in the first of these equations numbers the estimate of the power, not the signal whose power we refer to. We then calculate the correlation due to the magnitude quantization as

$$\gamma_{magQuant} = \frac{\sqrt{\mathrm{E}\left(s^2\right)_1}\cdot\sqrt{\mathrm{E}\left(s^2\right)_2}}{\sqrt{\mathrm{E}\left(s^2\right)_1+\mathrm{E}\left(n_1^2\right)+N_{q1}}\cdot\sqrt{\mathrm{E}\left(s^2\right)_2+\mathrm{E}\left(n_2^2\right)+N_{q2}}} \tag{28}$$

Whenever necessary, we assume a uniform clutter region within which each signal s_i has Rayleigh-distributed magnitude with a mean value determined by the average RCS of the clutter. The Rayleigh parameter β is determined as

$$\beta = \left(\text{mean RCS}\right)\sqrt{\frac{2}{\pi}}\,. \tag{29}$$

5.3.1 Linear Magnitude Case

We now evaluate Equation (23) and the resulting correlation with N_r=1. In this case, each image is represented as

$$im_i = \left(|s_i| + n_{qi}\right) \cdot e^{j\phi_i} = s_i + n_{qi} \cdot e^{j\phi_i} \tag{30}$$

The additional noise introduced by the compression is easily separated as

$$n_{compression,i} = n_{qi} \cdot e^{j\phi_i} \tag{31}$$

We are interested in the noise power, which we can obtain as the mean of the square of the compression noise, or

$$N_{qi} = \mathrm{E}\left(\left|n_{compression,i}\right|^2\right) = \mathrm{E}\left(n_{qi}^2\right) \tag{32}$$

where E( ) denotes statistical expectation.

The quantization noise is uniform from -0.5k_{scale} to +0.5k_{scale}, so

$$N_{qi} = \frac{k_{scale}^2}{12} \tag{33}$$

5.3.2 Square Root of Magnitude Case

We now evaluate Equation (23) and the resulting correlation with N_r=2. Notice that for this and all following cases, we use $k_{scale} = 1$. In this case, each image is represented as

$$im_i = \left(\sqrt{|s_i|} + n_{qi}\right)^2 \cdot e^{j\phi_i} = s_i + 2 \cdot n_{qi} \cdot \sqrt{|s_i|} \cdot e^{j\phi_i} + n_{qi}^2 \cdot e^{j\phi_i} \tag{34}$$

The additional noise introduced by the quantization is easily separated as

$$n_{compression,i} = 2 \cdot n_{qi} \cdot \sqrt{|s_i|} \cdot e^{j\phi_i} + n_{qi}^2 \cdot e^{j\phi_i} \tag{35}$$

We are more interested in the noise power, which we can obtain as the mean of the square of the quantization noise, or

$$N_{qi} = \mathrm{E}\left(\left|n_{compression,i}\right|^2\right) = \mathrm{E}\left(4 \cdot n_{qi}^2 \cdot |s_i| + 4 \cdot n_{qi}^3 \cdot \sqrt{|s_i|} + n_{qi}^4\right) \tag{36}$$

We assume that the quantization noise is independent of the signal and has zero mean. The noise power then reduces to

$$N_{qi} = 4 \cdot \mathrm{E}\left(n_{qi}^2\right) \cdot \mathrm{E}\left(\left|s_i\right|\right) + \mathrm{E}\left(n_{qi}^4\right) \tag{37}$$

As derived in Appendices A.1 and A.2, the additional noise power reduces to

$$N_{qi} = \frac{1}{3} \cdot \beta \cdot \sqrt{\frac{\pi}{2}} + \frac{1}{80} \tag{38}$$

5.3.3 Cube Root of Magnitude Case

We now evaluate Equation (23) and the resulting correlation with N_r=3. In this case, each image is represented as

$$im_i = \left(\left(\sqrt[3]{\left|s_i\right|} + n_{qi}\right)^3\right) \cdot e^{j\phi_i} = \left(\left|s_i\right| + 3 \cdot n_{qi} \cdot \left|s_i\right|^{\frac{2}{3}} + 3 \cdot n_{qi}^2 \cdot \left|s_i\right|^{\frac{1}{3}} + n_{qi}^3\right) \cdot e^{j\phi_i} \tag{39}$$

The additional noise introduced by the quantization is easily separated as

$$n_{qi} = \left(3 \cdot n_{qi} \cdot \left|s_i\right|^{\frac{2}{3}} + 3 \cdot n_{qi}^2 \cdot \left|s_i\right|^{\frac{1}{3}} + n_{qi}^3\right) \cdot e^{j\phi_i} \tag{40}$$

We are interested in the noise power, which we can obtain as the mean of the square of the quantization noise. Again, we assume that the quantization noise is independent of the signal and has zero mean. The noise power then reduces to

$$N_{qi} = \mathrm{E}\left(\left|n_{qi}\right|^2\right) = \mathrm{E}\left(9 \cdot n_{qi}^2 \cdot \left|s_i\right|^{\frac{4}{3}} + 12 \cdot n_{qi}^4 \cdot \left|s_i\right|^{\frac{2}{3}} + n_{qi}^6\right) \tag{41}$$

Using the derivations in Appendices A.1 and A.2, this becomes

$$N_{qi} = \frac{3}{4} \cdot \left(2 \cdot \beta^2\right)^{\frac{2}{3}} \cdot \Gamma\left(\frac{5}{3}\right) + \frac{3}{20} \cdot \left(2 \cdot \beta^2\right)^{\frac{1}{3}} \cdot \Gamma\left(\frac{4}{3}\right) + \frac{1}{448} \tag{42}$$

5.3.4 Fourth Root of Magnitude Case

We now evaluate Equation (23) and the resulting correlation with N_r=4. In this case, each image is represented as

$$im_i = \left(\left(\sqrt[4]{\left|s_i\right|} + n_{qi}\right)^4\right) \cdot e^{j\phi_i} = \left(\left|s_i\right| + 4 \cdot n_{qi} \cdot \left|s_i\right|^{\frac{3}{4}} + 6 \cdot n_{qi}^2 \cdot \left|s_i\right|^2 + 4 \cdot n_{qi}^3 \cdot \left|s_i\right|^{\frac{1}{4}} + n_{qi}^4\right) \cdot e^{j\phi_i} \tag{43}$$

The additional noise introduced by the quantization is easily separated as

$$n_{qi} = \left(4 \cdot n_{qi} \cdot |s_i|^{\frac{3}{4}} + 6 \cdot n_{qi}^2 \cdot |s_i|^2 + 4 \cdot n_{qi}^3 \cdot |s_i|^{\frac{1}{4}} + n_{qi}^4 \right) \cdot e^{j\phi_i} \tag{44}$$

We are interested in the noise power, which we can obtain as the mean of the square of the quantization noise. Again, we assume that the quantization noise is independent of the signal and has zero mean. The noise power then reduces to

$$N_{qi} = \mathrm{E}\left(|n_{qi}|^2 \right) = \mathrm{E}\left(16 \cdot n_{qi}^2 \cdot |s_i|^{\frac{3}{2}} + 68 \cdot n_{qi}^4 \cdot |s_i| + 28 \cdot n_{qi}^6 \cdot |s_i|^{\frac{1}{2}} + n_{qi}^8 \right) \tag{45}$$

Using the derivations in Appendices A.1 and A.2, this becomes

$$N_{qi} = \frac{4}{3} \cdot \left(2 \cdot \beta^2 \right)^{\frac{3}{4}} \cdot \Gamma\left(\frac{7}{4} \right) + \frac{17}{20} \cdot \left(2 \cdot \beta^2 \right)^{\frac{1}{2}} \cdot \Gamma\left(\frac{3}{2} \right) + \frac{1}{16} \cdot \left(2 \cdot \beta^2 \right)^{\frac{1}{4}} \cdot \Gamma\left(\frac{5}{4} \right) + \frac{1}{2304} \tag{46}$$

5.3.5 Log of Magnitude Case

For the case where we take the log instead of a root of the magnitude, we proceed slightly differently from the above. We begin with the same assumptions. Then, instead of taking a root, we take the log,

$$t_i = \log\left(|s + n_i| \right) \tag{47}$$

We then quantize this value to the nearest integer. We also clip the highest-magnitude pixels by limiting them to 2^{Nm}-1 where $\mathrm{N_m}$ is the number of bits to store for the magnitude.

$$t_i^{'} = \log\left(|s_i| \right) + n_{qi} + n_{ci} \tag{48}$$

Notice that the quantization is very different when done in the log domain as compared to the linear magnitude domain. On the other hand, the action of the clipping function in the two domains is identical if we adjust parameters appropriately. Thus, we now assume that we do the clipping after returning to the magnitude domain. In the magnitude domain, we multiply by the phase of the original image to return to a complex image

$$im_i = \left(\exp\left(\log\left(|s_i| \right) + n_{qi} \right) + n_{ci} \right) \cdot e^{j\phi_i} = \left(|s_i| \cdot \exp\left(n_{qi} \right) + n_{ci} \right) \cdot e^{j\phi_i} \tag{49}$$

The additional noise term is easily approximated as its Taylor series,

$$\exp\left(n_{qi} \right) \approx 1 + n_{qi} + \frac{n_{qi}^2}{2} \tag{50}$$

We can obtain the noise power as the mean of the square of the quantization noise. We make the same assumptions on independence that we made in the previous sections. We also assume for the moment that there is no clipping. The noise power then reduces to

$$N_{qi} = \mathrm{E}\left(\left|n_{qi}\right|^2\right) = \mathrm{E}\left(\left|s_i\right|^2 \cdot \left(n_{qi} + \frac{n_{qi}^2}{2}\right)^2\right) = \mathrm{E}\left(\left|s_i\right|^2 \cdot \left(n_{qi}^2 + n_{qi}^3 + \frac{n_{qi}^4}{4}\right)\right)$$
$$N_{qi} = \mathrm{E}\left(\left|s_i\right|^2\right) \cdot \mathrm{E}\left(n_{qi}^2\right) + \frac{1}{4} \cdot \mathrm{E}\left(\left|s_i\right|^2\right) \cdot \mathrm{E}\left(n_{qi}^4\right) \tag{51}$$

Using the derivations in Appendices A.1 and A.2, this becomes

$$N_{qi} = \frac{1}{6} \cdot \beta^2 \cdot \Gamma\left(2\right) + \frac{1}{160} \cdot \beta^2 \cdot \Gamma\left(2\right) = \beta^2 \left(\frac{1}{6} + \frac{1}{160}\right) = \frac{83}{480}\beta^2 \tag{52}$$

5.3.6 Transform-Based Methods

In this study, we have examined two transform-based lossy compression methods: JPEG compression and quantizing the FFT of the complex image. Both of these methods are too complex to be examined theoretically in this document. We thus do not compare the trial results to any theoretical result.

5.3.7 Lloyd's Quantizer

Lloyd's quantizer is a data-dependent quantizer. There is nothing theoretical that can easily be said about the performance of compression based on this quantizer beyond the fact that the error is expected to be better than the corresponding uniform quantizer (if the parameters are chosen correctly). Thus, we will compare the cases using Lloyd's quantizer to the theoretical value for a uniform quantizer with the same non-linear operation, and expect to see better performance.

5.4 Comments on Magnitude Clipping

We now return to the clipping noise terms in Equation (23). For the moment, we ignore the quantization noise characterized in the previous section and focus only on the clipping. Using Equation (23) in Equation (25), we obtain

$$\gamma = \frac{\left|\sum\left(s_1 - n_{c1} \cdot e^{j\phi_1}\right)\left(s_2^* - n_{c2} \cdot e^{-j\phi_2}\right)\right|}{\sqrt{\sum\left|s_1 - n_{c1} \cdot e^{j\phi_1}\right|^2 \cdot \sum\left|s_2 - n_{c2} \cdot e^{j\phi_2}\right|^2}} \tag{53}$$

We will attempt to simplify this equation in two different ways and show the error in each.

5.4.1 Assume Independence

We assume for simplicity that the quantization noise and the clipping noise are mutually independent and are each independent with respect to s_1 and s_2. We also assume that the two images have identical noise characteristics. Then the correlation reduces to

$$\gamma = \frac{\left|\sum s_1 s_2^*\right|}{\sum |s_1|^2 + \sum n_{ci}^2} = \frac{\sum |s|^2}{\sum |s|^2 + \sum |n_1|^2 + \sum n_{ci}^2} \tag{54}$$

In beginning this discussion, we note that the clipping noise will in actuality be strongly correlated with the image it comes from (and thus with the other image as well for high SNR). Thus, the estimate of correlation obtained in this manner is expected to significantly underestimate real data.

We can rearrange Equation (54) so that the correlation is given in terms of the original correlation without magnitude quantization, as

$$\gamma = \frac{\dfrac{\sum |s|^2}{\sum |s|^2 + \sum |n_i|^2}}{1 + \dfrac{\sum n_{ci}^2}{\sum |s|^2 + \sum |n_i|^2}} = \frac{\gamma_{original}}{1 + \dfrac{\sum n_{ci}^2}{\sum |s|^2 + \sum |n_i|^2}} \tag{55}$$

$$\gamma_{magClipping,independent} = \frac{1}{1 + \dfrac{\sum n_{ci}^2}{\sum |s|^2 + \sum |n_i|^2}} = \frac{P_i}{P_i + \sum n_{ci}^2} \tag{56}$$

where P_i is the total (signal plus noise) power in the original images.

As a formal definition of n_{ci},

$$n_{ci}(s_i) = \begin{cases} 0 & s_i \le \left(\left(2^{N_m} - 1\right) \cdot k_{scale}\right)^{N_r} \\ \left(k_{scale} \cdot \left(2^{N_m} - 1\right)\right)^{N_r} - s_i & s_i > \left(\left(2^{N_m} - 1\right) \cdot k_{scale}\right)^{N_r} \end{cases} \tag{57}$$

Appendix A.3 derives the average power in the clipping noise as

$$\begin{aligned} \mathrm{E}\left(n_{ci}^2\right) = & -\sqrt{2 \cdot \pi}\beta \cdot k_{scale}^{N_r} \cdot \left(2^{N_m} - 1\right)^{N_r} \\ & + 2\left(k_{scale}^{2N_r}\left(2^{N_m} - 1\right)^{2N_r} + \beta^2\right) \exp\left(-\frac{k_{scale}^{2N_r}\left(2^{N_m} - 1\right)^{2N_r}}{2\beta^2}\right) \end{aligned} \tag{58}$$

For highly clipped cases, this value approaches the power in the original image (signal plus noise). Thus, the predicted correlation for these cases is close to 0.5, the correlation which corresponds to a signal-to-noise ratio of 0 dB. However, the actual correlation is only slightly lower than the case with no clipping. Figure 21 shows the predicted and actual effective SNR and correlation for a set of tests using 77 of the images pairs referred to in Section 1. In this example, we clip the magnitude, with no quantizing beyond the original 16-bit magnitude and phase image. For each image, we measure the actual noise introduced (the difference between the original image and the clipped image) and use this value to calculate the correlation that would be expected if this were independent random noise. The actual correlation is much higher than this predicted value, indicating that the noise thus introduced is highly correlated between the two images.

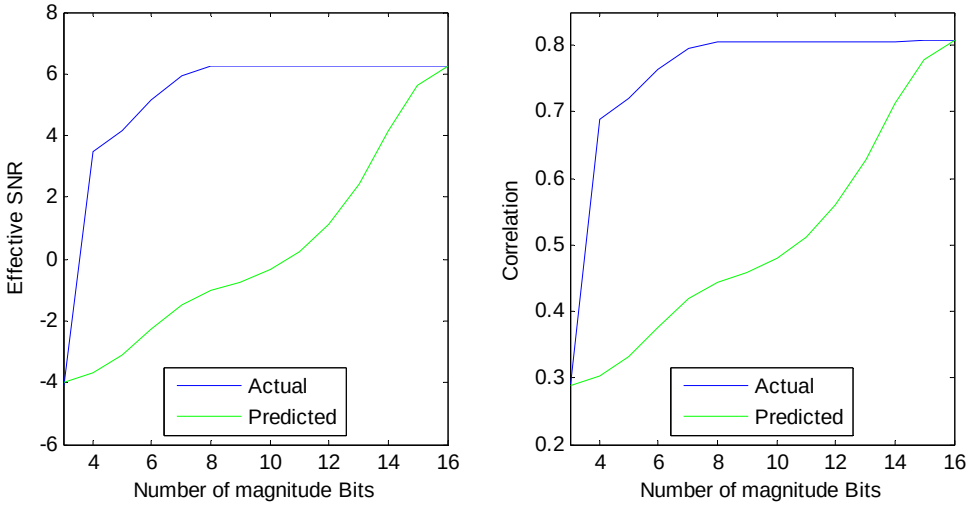

Figure 21. Predicted and actual correlation and effective SNR due to clipping the magnitude.

5.4.2 Assume Identical Correlation

Now, we return to Equation (53) but without the assumption of independence. Assuming the noise in the two images is independent and identically distributed, we can expand the multiplication and write the correlation as

$$\gamma = \frac{\sum s_1 \cdot s_2^* - \sum n_{c1} \cdot e^{j\phi_1} \cdot s_2^* - \sum s_1 \cdot n_{c2} \cdot e^{-j\phi_2} + \sum n_{c1} \cdot n_{c2} \cdot e^{j(\phi_1 - \phi_2)}}{\sum |s|^2 + \sum |n_1|^2 + \sum n_{c1}^2 - 2 \cdot \sum |s_1| \cdot n_{c1}} \tag{59}$$

We now assume that the correlation of the clipping noise from image 1 with image 2 is the same as the correlation between the original images. This allows us to separate the clipping noise and the image magnitudes into two different expectations. Similarly, we assume that the correlation between the clipping noise from the two images is the same as the correlation between the original images. We then write the entire correlation as

$$\gamma \approx \frac{\sum s_1 \cdot s_2^* - \gamma_{original} \sqrt{\sum n_{c1}^2 \cdot \sum |s_2|^2} - \gamma_{original} \sqrt{\sum n_{c2}^2 \cdot \sum |s_1|^2} + \gamma_{original} \sqrt{\sum n_{c1}^2 \cdot \sum n_{c2}^2}}{\sum |s|^2 + \sum |n_1|^2 + \sum n_{c1}^2 - 2 \cdot \sum |s_1| \cdot n_{c1}} \tag{60}$$

We then divide top and bottom by the original signal plus noise power, and factor out the original correlation from the entire expression.

$$\gamma \approx \gamma_{original} \left(\frac{1 + \sqrt{\frac{\sum n_{c1}^2}{\sum |s|^2 + \sum |n_1|^2}} + \sqrt{\frac{\sum n_{c2}^2}{\sum |s|^2 + \sum |n_1|^2}} + \frac{\sqrt{\sum n_{c1}^2 \cdot \sum n_{c2}^2}}{\sum |s|^2 + \sum |n_1|^2}}{1 + \frac{\sum n_{c1}^2 + 2 \cdot \sum |s_1| \cdot n_{c1}}{\sum |s|^2 + \sum |n_1|^2}} \right) \tag{61}$$

Now we again use the assumption that the noise terms from the two channels are identically distributed to simplify to

$$\gamma \approx \gamma_{original} \left(\frac{1 + 2\sqrt{\frac{\sum n_{c1}^2}{\sum |s|^2 + \sum |n_1|^2}} + \frac{\sum n_{c1}^2}{\sum |s|^2 + \sum |n_1|^2}}{1 + \frac{\sum n_{c1}^2 + 2 \cdot \sum |s_1| \cdot n_{c1}}{\sum |s|^2 + \sum |n_1|^2}} \right) \tag{62}$$

so that the correlation due to clipping the magnitude is approximated as

$$\gamma_{clipping,identicalCorrelation} \approx \left(\frac{1 + 2\sqrt{\frac{\sum n_{c1}^2}{\sum |s|^2 + \sum |n_1|^2}} + \frac{\sum n_{c1}^2}{\sum |s|^2 + \sum |n_1|^2}}{1 + \frac{\sum n_{c1}^2 + 2 \cdot \sum |s_1| \cdot n_{c1}}{\sum |s|^2 + \sum |n_1|^2}} \right). \tag{63}$$

These equations have a nice form: the original correlation multiplied by a new term that represents the correlation due to the clipping noise. However, they do not accurately predict the correlation seen in tests with real data. The reason for this is that both the numerator and denominator of Equation (59) contain several terms whose magnitude is large but whose sum is much smaller than any individual term. Errors in approximating such values are magnified. For one image tested, the individual terms were on the order of 10^7, while the sum was on the order of 10^4. With such a sum, a 1% error in approximating one of the terms has the potential to cause a 900% error in the sum. While it is not known which is the most significant cause of the error, we have introduced the following approximations:

- Assume that the correlation between clipping noise and an image is the same as the correlation between the two images.

- Assume that the image magnitudes have a Rayleigh distribution (the real images examined in this study are significantly different from this).

5.5 The Effect of Adding Independent Gaussian Noise

In our studies to determine why the correlation predictions in the previous subsection are so inaccurate, we added independent Gaussian noise to the image pairs to confirm that this type of noise affects the correlation in the expected way. We discovered that the image pairs from the November 1, 2004 flight test do not behave as expected in the presence of added Gaussian noise.

One of the reasons for all of this theoretical work is to confirm that the results reported in Section 1 are valid even though the data used to generate most of those results is suspect. The correlation in these image pairs is significantly smaller than would be expected given the actual estimated signal-to-noise ratio. Some amount of work has gone into the problem of determining why the correlation is so low, but to our knowledge no conclusions have been reached as of this writing. We include this information in part to help in that effort.

Given the image and signal models in Section 5.3, the theoretical correlation after adding noise is

$$\gamma_{gaussian} = \frac{S}{S+N+N_1} = \frac{\frac{S}{S+N}}{1+\frac{N_1}{S+N}} = \gamma_{original} \cdot \frac{P_{tot}}{P_{tot}+N_1} \tag{64}$$

where the capital letters indicate average power in each signal, and P_{tot} is the original signal-plus-noise power. Notice that the expected correlation is independent of any assumptions on signal-to-noise ratio and only depends on the original correlation, the original image power, and the additional noise power.

Figure 22 shows the results of a set of tests where we added independent noise with Gaussian real and imaginary parts to each image in a pair, then calculated the new correlation. The different colors of dots represent different amounts of noise added to the original images, and the blue lines represent the points where the real and theoretical values match. The left two plots show the results from a set of 22 IFSAR image pairs in the Buck Well region. The right two plots show the results from 86 of the CCD image pairs from the November 1, 2004 flight. The top plots show the effective SNR, and the bottom plots show the correlation.

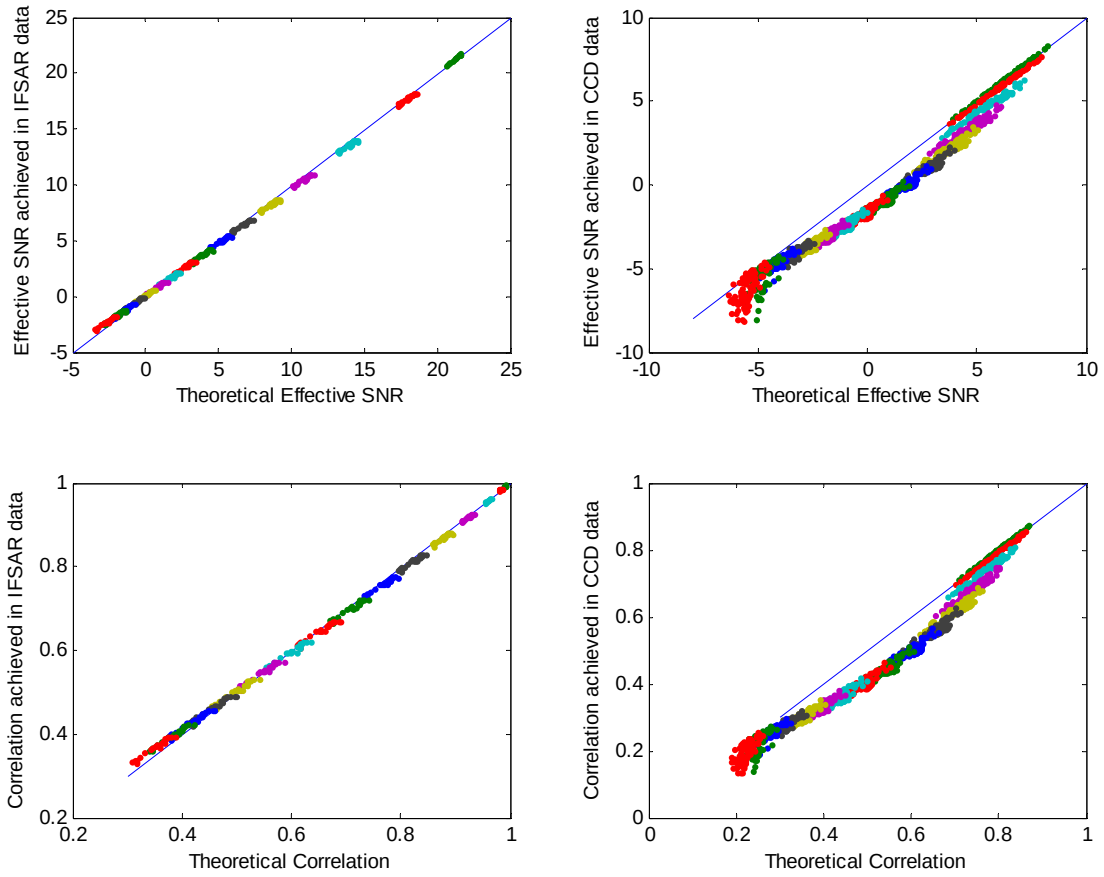

Figure 22. Comparing theory to reality for the CCD correlation and the IFSAR correlation.

Notice that the IFSAR data responds to the added noise as expected, but the CCD data loses correlation more quickly than we would expect. The "tails" extending down from the left edge of the CCD data are due to cases where registration errors have been introduced. To be sure that these are the only cases where registration is an issue, we ran a similar set of tests with the registration held constant for all cases of a particular image. The results were identical to those in Figure 22 except for the tails.

5.6 Comparing Theory to Reality

We now combine the theoretical results of Sections 5.2 and 5.3 to obtain expected correlation values corresponding to the values achieved in Sections 2-1. The results of the comparison are shown in Table 7. For each lossy compression scheme, we show the predicted correlation, the actual average correlation, and the difference (actual minus predicted). We recall that we did not derive theoretical correlations for any of the transform-based methods, so we leave them out of the comparison. We also do not have a good prediction for correlation in the presence of clipping, so we assume in all cases that there is no clipping. The predicted values shown in the cases with Lloyd's quantizer are actually the values for the square-root uniform quantizer with the same numbers of bits. We would expect Lloyd's quantizer to perform better.

Table 7. Comparing the predicted correlations to the actual correlations for the tests in Sections 2-1

	Predicted	Actual	Difference
16-bit mag,16-bit phase	1.0000	1.0000	0.0000
8-bit mag^(1/2),12-bit phase	0.9963	0.9966	0.0003
8-bit mag^(1/2),8-bit phase	0.9962	0.9966	0.0003
9-bit mag/2, 4-bit phase	0.9871	0.9898	0.0027
8-bit mag/2, 4-bit phase	0.9871	0.9896	0.0025
8-bit mag^(1/2),6-bit phase	0.9955	0.9960	0.0005
8-bit mag/4, 4-bit phase	0.9869	0.9895	0.0026
8-bit 95% jpeg,8-bit phase	--	0.9474	--
5-bit mag^(1/2),6-bit phase	0.9955	0.9952	-0.0003
6-bit mag^(1/2),6-bit phase	0.9955	0.9955	-0.0000
7-bit mag/8, 4-bit phase	0.9861	0.9880	0.0019
6,6-bit I/Q of FFT	--	0.9931	--
8-bit 90% jpeg,8-bit phase	--	0.9250	--
4-bit mag^(1/2),6-bit phase	0.9955	0.9931	-0.0024
8-bit 80% jpeg,8-bit phase	--	0.8872	--
4-bit mag^(1/3),6-bit phase	0.9964	0.9690	-0.0274
8-bit mag^(1/2),4-bit phase	0.9835	0.9870	0.0035
8-bit 95% jpeg,6-bit phase	--	0.9469	--
6-bit mag^(1/2),4-bit phase	0.9835	0.9865	0.0030
5-bit mag^(1/2),4-bit phase	0.9835	0.9862	0.0027
4,4-bit I/Q FFT, 2 stds	--	0.9404	--
3-bit Lloyd's, 5-bit phase	0.9931	0.9780	-0.0151
6-bit Lloyd's, 3-bit phase	0.9451	0.9594	0.0144
4-bit Lloyd's, 4-bit phase	0.9835	0.9837	0.0002
4,4-bit I/Q FFT, 1 stds	--	0.7883	--
8-bit 90% jpeg,6-bit phase	--	0.9245	--
5-bit Lloyd's, 3-bit phase	0.9451	0.9578	0.0127
4-bit mag^(1/2),4-bit phase	0.9835	0.9841	0.0006
5-bit mag^(1/2),3-bit phase	0.9451	0.9577	0.0126
4,4-bit I/Q FFT, 3 stds	--	0.9690	--
3-bit ln mag,5-bit phase	0.9306	0.9429	0.0123
3-bit mag^(1/4),4-bit phase	0.8706	0.9063	0.0357
3-bit ln mag,4-bit phase	0.9216	0.9361	0.0144
4-bit mag^(1/3),3-bit phase	0.9460	0.9323	-0.0136
3-bit mag^(1/4),5-bit phase	0.8791	0.9130	0.0338
4-bit mag^(1/3),4-bit phase	0.9844	0.9602	-0.0242
4,4-bit I/Q FFT, 4 stds	--	0.9560	--
8-bit 95% jpeg,4-bit phase	--	0.9384	--
8-bit mag^(1/2),2-bit phase	0.7914	0.8499	0.0585
6-bit mag^(1/2),2-bit phase	0.7914	0.8494	0.0580
4,4-bit I/Q of FFT	--	0.9081	--
3-bit ln mag,3-bit phase	0.8857	0.9089	0.0233
3-bit mag^(1/4),2-bit phase	0.7006	0.7799	0.0793
2-bit mag^(1/4),1-bit phase	0.1566	0.5104	0.3539

Notice that the predicted and actual values are fairly close for all cases. In general, the difference between them increases as the number of bits decreases. This is expected because our small-value and independent noise assumptions become less valid. Notice that the 4-bit magnitude, 4-bit phase case with Lloyd's quantizer is slightly worse than the predicted value and slightly worse than the corresponding case with the square-root uniform quantizer. This is one case where multiple performance measures are required – the uniform quantized case is not usable because of the extreme clipping of any bright pixels in the magnitude image, but the case with Lloyd's quantizer has a usable

magnitude image and only slightly worse correlation. Lloyd's quantizer is better than the listed predicted value for those cases with 5 or more bits of magnitude, but worse for cases with fewer bits. Perhaps this is because the clipping noise introduced by the uniform quantizer with few bits has less effect on the correlation than the quantization noise introduced by Lloyd's quantizer in the same cases. Referring back to Table 6, at 4 bits magnitude, 4 bits phase, Lloyd's quantizer is slightly worse than the uniform quantizer for all measures of performance except the RMS of the difference in the correlation map, which is slightly better. The RMS difference is more strongly affected by the error the clipping causes around bright points, compared to the other measures which tend to average out such errors.

6 Conclusions and Recommendations

This document presented the results of a series of tests examining over 600 CCD image pairs using 44 different lossy compression schemes. We also derived theoretical predictions for the correlation for most of these compression schemes. The results presented here are averages over many different image pairs, so some cases can be expected to be significantly better or worse. Given all this information, this section gives recommendations for future CCD projects and for future work on the present topic.

6.1 Recommendations for Limited-Bandwidth CCD Programs

We make the following suggestions for programs which desire to perform CCD and which have limited image bandwidth:

- Use some form of lossless compression. While we have not done a detailed study on the relative effectiveness of the various lossless compression schemes, zlib seems to provide a good balance between simplicity, compression time, and compression ratio. This library is available in source form and is easily portable (for Tactical IFSAR, it was compiled for at least Mercury, Solaris, Windows, and Linux).

- For projects which require no loss in CCD performance, use 8-bit square-root of magnitude and 6-bit phase data.

- For projects which can allow a small degradation in CCD performance, use 8-bit square-root of magnitude and 4-bit phase data.

- For projects which require an 8-bit pixel at the cost of a larger degradation in CCD performance, use Lloyd's quantization scheme with 4 bits on the square root of the magnitude and use 4-bit phase data. With zlib compression, this scheme is only slightly smaller than 8-bit square-root of magnitude and 4-bit phase data, but without lossless compression there is a more significant size difference.

6.2 Recommendations for Future Work on this Topic

- Complete the theoretical analysis: magnitude clipping, transform-based methods, and Lloyd's quantizer.

- Try some image pairs with more bright (high reflectivity) stuff in the image. We do have some corner reflectors and vehicles in these images, which may be good enough. The biggest difference would probably be that images with a lot of bright regions would not compress as small in the lossless step.

7 References

1. Lloyd, S. P., “Least Squares Quantization in PCM”, IEEE Transactions on Information Theory, Vol. IT-28, No. 2, pp. 129-137, Mar. 1982.

2. Eichel, P, Ives, RW, “Compression of complex-valued SAR images”. IEEE Tansactions on Image Processing; Oct. 1999; vol.8, no.10, p.1483-7.

3. \\sass4800\dgthomp\matlab\cpackunpack.c

4. http://www.zlib.org

Appendix A A few statistical derivations

A.1 Expectations of a Rayleigh Random Variable

Many of the theoretical formulas for correlation derived in Section 1 include powers of the image magnitude, which we model as a Rayleigh random variable. We here derive the expectation of expressions of the form x^a where a is positive and x is a Rayleigh-distributed random variable. The probability density function for such variables is

$$f(x) = \frac{x}{\beta^2} \cdot \exp\left(\frac{-x^2}{2\beta^2}\right), 0 \le x \tag{65}$$

The desired expectation is then

$$\mathrm{E}\left(x^a\right) = \int_0^\infty x^a \cdot \frac{x}{\beta^2} \cdot \exp\left(\frac{-x^2}{2\beta^2}\right) dx \tag{66}$$

We make the substitution

$$\begin{aligned} t &= \frac{x^2}{2\beta^2} \\ x &= \beta \cdot \sqrt{2t} \\ dx &= \frac{\beta}{\sqrt{2t}} dt \end{aligned} \tag{67}$$

resulting in

$$\mathrm{E}\left(x^a\right) = \int_0^\infty \frac{\left(\beta \cdot \sqrt{2t}\right)^{a+1}}{\beta^2} \cdot \exp(-t) \cdot \frac{\beta}{\sqrt{2t}} \cdot dt = 2^{\frac{a}{2}} \beta^a \int_0^\infty t^{\frac{a}{2}} \cdot \exp(-t) \cdot dt \tag{68}$$

We note that the integral is in the form of the gamma function,

$$\Gamma(z) = \int_0^\infty t^{z-1} \exp(-t)\, dt \tag{69}$$

So our expectation can be written in terms of the gamma function as

$$\mathrm{E}\left(x^a\right) = 2^{\frac{a}{2}} \beta^a \cdot \Gamma\left(\frac{a}{2} + 1\right) \tag{70}$$

A.2 Expectations of a Uniform Random Variable

Our theoretical correlations require expectations of several functions of a uniform random variable. For all cases, the uniform random variable we use has a lower limit equal to the negative of the upper limit. For most cases, these limits are ±0.5. We use these assumptions to help simplify the derivations. Again, we will find the expectation of an expression x^a where x is now uniformly distributed. The probability density function is now

$$f(x) = \frac{1}{\beta - \alpha}, \alpha < x < \beta \tag{71}$$

Our integral is then

$$\mathrm{E}\left(x^a\right) = \int_{\alpha}^{\beta} \frac{x^a}{\beta - \alpha} = \left. \frac{x^{a+1}}{(a+1)(\beta - \alpha)} \right|_{\alpha}^{\beta} \tag{72}$$

We simplify at this point by recalling that we have said that $\beta = -\alpha$.

$$\mathrm{E}\left(x^a\right) = \left. \frac{x^{a+1}}{(a+1)(-\alpha - \alpha)} \right|_{\alpha}^{-\alpha} = \frac{(-\alpha)^{a+1} - (\alpha)^{a+1}}{-(a+1)(2\alpha)} \tag{73}$$

If a is odd, the numerator cancels and the expectation is zero. If a is even, the expression simplifies to

$$\mathrm{E}\left(x^a\right) = \frac{2\alpha^{a+1}}{(a+1)(2\alpha)} = \frac{\alpha^a}{(a+1)} \tag{74}$$

A.3 Average Power in Rayleigh Clipping Noise

We now calculate the average power in the noise introduced by clipping the Rayleigh-distributed magnitude. The clipping noise is repeated here from Equation (57).

$$n_{ci}(s_i) = \begin{cases} 0 & s_i \le \left(\left(2^{N_m} - 1\right) \cdot k_{scale}\right)^{N_r} \\ \left(k_{scale} \cdot \left(2^{N_m} - 1\right)\right)^{N_r} - s_i & s_i > \left(\left(2^{N_m} - 1\right) \cdot k_{scale}\right)^{N_r} \end{cases} \tag{75}$$

The integral here has a different form compared to Equation (66) only by the lower limit.

$$\mathrm{E}\left(n_{ci}^2\right) = \int_{-\infty}^{\infty} f(x) n_{ci}^2(x) dx = \int_{\left(2^{Nm} - 1\right)^{N_r} \cdot k_{scale}^{N_r}}^{\infty} \frac{x}{\beta^2} \exp\left(\frac{-x^2}{2\beta^2}\right) \left(k_{scale}^{N_r} \cdot \left(2^{N_m} - 1\right)^{N_r} - x\right)^2 dx \tag{76}$$

This simplifies to

$$\mathrm{E}\left(n_{ci}^2\right) = \int_{\left(2^{Nm}-1\right)^{N_r}\cdot k_{scale}^{N_r}}^{\infty} \frac{x}{\beta^2}\exp\left(\frac{-x^2}{2\beta^2}\right)\left(k_{scale}^{2N_r}\cdot\left(2^{N_m}-1\right)^{2N_r} - 2\cdot x\cdot k_{scale}^{N_r}\left(2^{N_m}-1\right)^{N_r} + x^2\right)dx \quad (77)$$

We rewrite this as

$$\mathrm{E}\left(n_{ci}^2\right) = \int_a^{\infty}\left(b\cdot x + c\cdot x^2 + d\cdot x^3\right)\cdot\exp\left(-e\cdot x^2\right)dx \quad (78)$$

where we define the constants

$$\begin{aligned} a &= \left(2^{Nm}-1\right)^{N_r}\cdot k_{scale}^{N_r} \\ b &= \frac{k^{2N_r}\cdot\left(2^{Nm}-1\right)^{2N_r}}{\beta^2} \\ c &= -\frac{2k_{scale}^{N_r}\left(2^{N_m}-1\right)^{N_r}}{\beta^2} \\ d &= \frac{1}{\beta^2} \\ e &= \frac{1}{2\beta^2} \end{aligned} \quad (79)$$

We can integrate the parts of Equation (78) separately.

$$\int_a^{\infty} b\cdot x\cdot\exp\left(-ex^2\right)dx = -\frac{b}{2e}\cdot\exp\left(-ex^2\right)\Big|_a^{\infty} = \frac{b}{2e}\cdot\exp\left(-ea^2\right) \quad (80)$$

$$\int_a^{\infty} d\cdot x^3\cdot\exp\left(-e\cdot x^2\right)dx = \frac{d\cdot a^2}{2\cdot e}\cdot\exp\left(-ea^2\right) + \frac{d}{2\cdot e^2}\cdot\exp\left(-ea^2\right) \quad (81)$$

$$\int_a^{\infty} c\cdot x^2\cdot\exp\left(-e\cdot x^2\right)dx = \frac{c\cdot a}{2\cdot e}\cdot\exp\left(-ea^2\right) + \frac{c}{2\cdot e}\cdot\frac{1}{2}\sqrt{\frac{\pi}{e}}\,\mathrm{erfc}\left(a\sqrt{e}\right) \quad (82)$$

Here erfc represents the complementary error function, or one minus the error function. We can approximate this function in the cases where its argument is especially large or small. The small-value approximation is

$$\int_a^{\infty} c\cdot x^2\cdot\exp\left(-e\cdot x^2\right)dx = \frac{c\cdot a}{2\cdot e}\cdot\exp\left(-ea^2\right) + \frac{c}{2\cdot e}\cdot\frac{1}{2}\sqrt{\frac{\pi}{e}} - \frac{c\cdot a}{2\cdot e}\cdot\exp\left(-ea^2\right) = \frac{c}{2\cdot e}\cdot\frac{1}{2}\sqrt{\frac{\pi}{e}} \quad (83)$$

The large-value approximation is

$$\int_a^{\infty} c \cdot x^2 \cdot \exp\left(-e \cdot x^2\right) dx = \frac{c \cdot a}{2 \cdot e} \cdot \exp\left(-ea^2\right) + \frac{c}{2 \cdot e} \cdot \frac{1}{2} \sqrt{\frac{\pi}{e}} - \frac{c}{4 \cdot a \cdot e^2} \cdot \exp\left(-ea^2\right) \quad (84)$$

Finally, we arrive at the result by combining the terms and simplifying

$$\mathrm{E}\left(n_{ci}^2\right) = \begin{cases} \frac{c}{4 \cdot e} \cdot \sqrt{\frac{\pi}{e}} + \frac{\exp\left(-ea^2\right)}{2 \cdot e}\left(b \cdot + d \cdot a^2 + \frac{d}{e}\right) & a\sqrt{e} \square\ 1 \\ \frac{\exp\left(-ea^2\right)}{2 \cdot e}\left(b \cdot + c \cdot a + \frac{c}{2 \cdot a \cdot e} + d \cdot a^2 + \frac{d}{e}\right) & a\sqrt{e} \square\ 1 \end{cases} \quad (85)$$

If we now insert the values for our constants *a*, *b*, *c*, *d*, and *e*, we arrive at

$$\begin{aligned} \mathrm{E}\left(n_{ci}^2\right) = & -\sqrt{2 \cdot \pi}\beta \cdot k_{scale}^{N_r} \cdot \left(2^{N_m} - 1\right)^{N_r} \\ & + 2\left(k_{scale}^{2N_r}\left(2^{N_m} - 1\right)^{2N_r} + \beta^2\right)\exp\left(-\frac{k_{scale}^{2N_r}\left(2^{N_m} - 1\right)^{2N_r}}{2\beta^2}\right) \end{aligned} \quad (86)$$

for small values of $a\sqrt{e}$ and $\mathrm{E}\left(n_{ci}^2\right) = 0$ otherwise. For large values of $a\sqrt{e}$, there is no clipping, which explains why the clipping noise goes away. Notice that for any pixels which are clipped, the quantization noise is included in both n_{ci} and n_{qi}. However, for such cases, the clipping noise dominates the quantization noise, so adding a small amount of additional quantization noise should have little effect.

DISTRIBUTION

Unlimited Release

1	MS 1330	B. L. Remund	2340
1	MS 0519	B. L. Burns	2340
1	MS 0519	W. H. Hensley	2342
1	MS 0519	T. P. Bielek	2342
1	MS 1330	A. W. Doerry	2342
1	MS 0519	D. W. Harmony	2342
1	MS 0519	J. A. Hollowell	2342
1	MS 0519	S. S. Kawka	2342
1	MS 1330	M. S. Murray	2342
6	MS 0519	D. G. Thompson	2342
1	MS 0529	K. W. Sorensen	2345
1	MS 0529	D. F. Dubbert	2345
1	MS 0529	G. R. Sloan	2345
1	MS 1330	S. M. Becker	2348
1	MS 0519	S. M. Devonshire	2348
1	MS 0519	M. T. Gardner	2348
1	MS 0519	L. M. Wells	2354
1	MS 0519	D. L. Bickel	2354
1	MS 0519	J. T. Cordaro	2354
1	MS 0519	J. M. Delaurentis	2354
1	MS 1207	C. V. Jakowatz, Jr.	5937
1	MS 1207	N. E. Doren	5937
1	MS 1207	P. H. Eichel	5937
1	MS 9018	Central Technical Files	8945-1
2	MS 0899	Technical Library	9616
		DOE ARS	DOE NA-22