SIMULATED PERFORMANCE RESULTS OF THE OMV VIDEO COMPRESSION TELEMETRY SYSTEM

Frank Ingels
Mississippi State University

Glenn Parker Lee Ann Thomas
Marshall Space Flight Center, NASA

ABSTRACT

The Orbital Maneuverable Vehicle (OMV) will use a man-in-the-loop round trip space-to-earth communication link for remote control and docking with an orbiting spacecraft. The control system uses range/range rate radar, a forward command link, and a compressed video return link. Figure 1 illustrates the overall compressed video coding techniques.

Analog RS-170 compatible video is available from any one of eight or, at a lower resolution, simultaneously from any two television cameras. The video data is digitized and then compressed by sampling every sixth frame of data. A rate of five frames per second is adequate for the OMV docking speeds. Further compression, at the expense of spatial resolution, is obtained by averaging adjacent pixels. The remaining compression is achieved using differential pulse code modulation (DPCM) and Huffman run length encoding. To protect this compressed video data stream from Space to TDRSS channel errors, a concatenated error correction coding system will be used. This concatenated coding is achieved by encoding with a helical interleaved (depth 8) Reed-Solomon (255,239) block code and then encoding with a rate 112 convolution code (constraint length 7) followed by a periodic convolution interleaver (30,116). Thus, we see that four stages of compression, two types of error correction encoding and two levels of interleaving are utilized in this fairly sophisticated data transmission system. A detailed system description and simulated system performance results are presented in this paper.

COMPRESSION CODING

The video camera output is an RS 170 standard, 525 lines per frame, 30 frames per second signal with a 510 by 488 pixel array. Assuming the use of 8 bits per pixel, a channel bandwidth of approximately 60 MHz would be required. Since the available bandwidth is only 486 KHz per channel, a compression ratio of 124:1 is necessary.

Prior to data compression, the original 8-bit pixel values are re-scaled by a digital compander. The compander is described by an equation for the output in terms of the input x by

$$\gamma = -153x^2 + 1.6x$$

The effect of the compander is to push the coding error into the higher luminescence ranges where it is not as easily detected by the eye (Reference 1). A combination of techniques are used to compress the data rate: reduced frame rate, pixel pairing, DPCM and Huffman run length codes. At the planned docking rate, analysis has shown that 5 frames per second is satisfactory, resulting in a 6:1 data compression. Further compression is obtained by averaging adjacent pixels (pixel pairing). This pixel averaging technique (which achieves up to a 4:1 compression), in combination with the reduced frame rate, provides a total data rate reduction ratio of 24:1. The remaining compression of 124/24 (5.16:1) is achieved using differential pulse code modulation (DPCM) and Huffman run length codes. The differential encoding operation uses the companded pixel values. By quantizing the differences in a fashion related to the probability density function of the grey scale value, the value of the difference between pixels may be transmitted by using a reduced, finite set of the possible differences (Reference 1). Thus, an effective reduction in data rate is achieved.

Variables in the DPCM unit are the probability density function for the grey scale and the choice of the differential step size in the DPCM's non-linear quantizer. The probability density function (PDF) is derived from histogram analysis of representative scenes, provided by NASA, of docking-type operations in space (Reference 2). It has been determined that 16 basic PDF's (sets of bins) will be sufficient for OMV docking scenes. The boundaries of these bin sets are determined by the quantized step (scaler) value. The quantized step value is adjusted by monitoring the amount of data in the output rate buffer. This creates an adaptive compression loop in which the level of compression is determined by the scene being transmitted.

The DPCM process allows an estimate of the next sample to be "predicted", using the values of neighboring pixels. This is accomplished by adaptive predictor tests that check a 2 by 2 array for a certain set of conditions, essentially vertical and horizontal edge detection. (Other algorithms included in adaptive predictor tests include the following: the first pixel of the frame or sub-frame is the absolute pixel value, the first pixel of each line uses only vertical pairing, and the first line of a frame or sub-frame uses only horizontal pixel pairing.) This predicted value is then subtracted from the actual value, then mapped into bin numbers by the non-uniform quantizer. This bin number along with the scaler value identifies the pixel. The next step is to check these bin numbers for two occurrences: bin numbers of value 0 for 10 or more consecutive times and bin numbers of values 0, +1 and -1 for four consecutive times. If either of these two cases is met, these are transmitted

with one single Huffman code word. All others will have pre-assigned Huffman code words, the most probable occurrences having been assigned the shortest Huffman codes (Reference 3). A separate Huffman table is utilized for each of a group of step sizes in the DPCM quantizer.

The bandwidth reduction achieved by this compression process does not have an observable effect on the reconstructed video data for the slow motion or no motion scenes anticipated for a docking OMV and target spacecraft. The closing rate of the OMV at docking is estimated to be 200 feet or less per minute.

DATA FORMATTER

Elimination of redundancy from the transmitted picture results in the compressed video return link being highly susceptible to errors induced by the channel environment. Errors thus induced will propagate until a new pixel compression reference is established. Propagation of channel induced errors is limited by the subdivision of a video frame into subframes, each containing header information which re-establishes the compression reference. Subframe sizes are command selectable and contain 4, 10, or 20 lines of video data. Each subframe contains a syncword, subframe I.D., reference pixel, scale factor for DPCM quantizer step size and two lines of video data. The first subframe of each video frame also contains a time tag and mode identification.

Since Huffman codes have variable length, the output of the Huffman coder has a variable rate, while the required channel rate is fixed. Rate matching is provided as an output rate buffer which is also the sensing point for the adaptive compression loop scalar generation. When the buffer reaches approximately half-full point, the scalar value is increased, resulting in a larger step size, thus coarser quantization, and reducing the number of bits per pixel. The output rate buffer bit stream is formatted for subsequent Reed-Solomon encoding into a 238-byte (1904 bits) buffer. A Reed-Solomon code word contains 238 information bytes (symbols). If all 238 bytes have not been filled when the R/S coder is ready to accept a new code word, a bit buffer function supplies the required fill bits. Channel interleaved data from the two R/S data field buffers are alternately transferred to the Reed-Solomon encoder, and a 239-byte is added to each word for synchronization purposes.

CODING

Figure 2 illustrates a top-level diagram of the coding techniques used in the compressed video return link to counter the effects of both random and burst errors. As shown in the diagram, at the last stage of the VCU, the compressed video data is Reed/Solomon encoded and helically interleaved. The data is then sent from the VCU to the Central Unit

in the Data Handling Equipment where it is convolutionally encoded (rate 1/2; constraint length 7) and periodically convolutionally interleaved. After application to the TDRSS User Transponder, the RF signal will be return-linked to the WSGT via the TDRSS. Ground-based RFI interference will be randomly experienced by the RF signal between the OMV and the TDRSS. After demodulation and bit synchronization at the WSGT, the baseband signal is convolutionally deinterleaved and Viterbi decoded. The resultant data stream is then transmitted to the OMV (GCC) at JSC via the NASCOM network. The baseband received data is clocked into the VRU where it is helically deinterleaved and Reed/Solomon decoded. This portion of the paper describes only those features unique to the OMV coding system, namely the Reed/Solomon coding, decoding and helical interleaving, deinterleaving and error containment processing implemented in the video compression and reconstruction processes. The convolutional encoding, Viterbi decoding and periodic convolutional interleaving and deinterleaving are features common to all TDRSS users having sufficiently high data rates.

The Reed/Solomon code selected for use on the OMV is a (255, 238) codeword which consists of 238 information symbols, 16 parity symbols and one synchronization symbol which is appended to the end of the codeword. Each symbol is an 8-bit word from a 256 symbol alphabet. The Reed/Solomon codeword as defined can be used to detect and correct up to eight symbol errors.

Since an erroneous symbol may contain a single bit error or as many as eight bits in error; a single codeword has the capability to correct as few as eight bit errors (each error located in a different symbol) or as many as 64 bit errors if all errors were located in exactly 8 symbols. Thus, nine errors spread over nine symbols would be uncorrectable. The code would, however, be capable of detecting that the number of errors exceeds the correction capability of the code.

The power of the Reed/Solomon codes are found in the correction of burst errors (rather than random or isolated bit errors) using a very low code overhead. The overhead for the (255, 238) codeword is only 7% (i.e. , 17/238 including sync), contrasted with the rate 1/2 convolutional code with 100% overhead. The convolutional codes, however, have a superior random error correction capability.

A depth 8 helically interleaved data structure (code block) will be used for OMV. Data from the source is encoded vertically into each codeword. The codewords are formed from right to left with each succeeding codeword being displaced vertically downward by 32 bytes. The data is then read into the channel from the left by rows. Thus, a staggered column in, row out structure is created. Alternation of the two video channels with the codewords provides an additional level of interleaving which provides for the spreading of burst. The helically interleaved structure provides the same random and burst error

correcting capability as the more commonly used block interleaved structure (eight symbols in a codeword, 64 symbols in a codeblock, and burst up to 505 bits in length). The advantage of helical versus block interleaved structures are:

1) less memory is required (one buffer)
2) less latency is introduced in the data before being read into the channel.

Another feature of helical interleaving and, perhaps, the most important from the standpoint of the uncertainty (i.e. , future increased degradation) associated with the RFI environment, is that the helical interleaving structure can permit the use of burst forecasting. Burst forecasting can extend the burst correction capability of the Reed/Solomon error correction system by up to a factor of two. Burst forecasting is not at present a requirement of the system, but may be in the future. If indeed at a later time, burst forecasting does become a requirement due to better definition of the current environment or future worsening of the environment, it is only necessary to modify the Reed/Solomon decoder on the ground. It is not necessary to redesign or modify the spaceborne VCU-Reed/Solomon encoder.

CODEWORD SYNCHRONIZATION

Inherent in the design of any asynchronous bit serial PCM data system is a requirement to identify boundaries of the encoded data such that reliable synchronization acquisition is possible in the decoder. As noted earlier, the (255,239) Reed-Solomon codeword is actually a (254,238) codeword extended to include a fixed 8-bit syncword. The nature of the helical interleaving technique used is such that there is a fixed distance, or phase relationship, between syncwords regardless of the depth of interleave. Thus, when the interleaved data is read into the channel a sync acquisition block of 255 symbols, or 2040 bits, is created. Data to the Reed-Solomon decoder is input to an 8-bit pattern detector providing an 8-bit window which slides along the received data string. Coincident with each bit time a modulo 2040 score address counter is incremented. This provides an address for each of 2040 sync score memories corresponding to 2040 possible block boundaries for a codeword. The addressed sync score memory is updated by the output of an adder which is incremented by one when the window matches a preset sync pattern and is decremented by one when there is no match. As this process continues, one of the sync score memories will count up, and the others will count down or remain at zero, absent ‘malicious’ data. Sync score memories incremented by malicious data would be decremented on the next cycle unless it were to occur consistently at 2040 bit intervals. When any given sync score is greater than the preset threshold, successful sync acquisition is declared. The synch acquisition block boundary is defined by a modulo 2040 phase counter which is reset when sync is acquired. Phase zero indicates the beginning of a new sync acquisition block. Following the initial location of a code block boundary,

synchronization is not ‘lost’ in the normal sense. Occasionally, a syncword will be corrupted by random or burst errors. In this case, the sync score memory will be decremented by one, however, sync is still valid. In the event of a bit slip, the codeword boundary previously defined will remain until another sync score memory address counts up to the preset threshold value. Thus, sync acquisition is not “lost”, but its phase is redefined. Critical to this process is the maximum value to which the sync score memory must be incremented. Presently, tests are being conducted to optimize this value. Tradeoff is obviously between the time to acquire synchronization and the probability of a false indication.

VIDEO SUBFRAME STRUCTURE AND SUBFRAME REPLACEMENT

Although the Reed/Solomon encoding and interleaving provide a very capable error detection and correction system, the problem of error propagation persists if uncorrectable errors were to remain in the compressed video data stream and error containment were not provided.

Two types of errors are in issue:

1) Those errors introduced in the channel subsequent to error correction coding which exceed the code correction capability and

2) Those errors which could be introduced by noise in the VCU before being Reed/Solomon encoded.

Residual uncorrectable errors would, however, be detected by the Reed/Solomon decoder and a flag provided to the downstream video reconstruction logic to implement corrective action. Errors in the second category above would never be detected by the error correction decoder. In this instance, other mechanisms are in place to detect this situation and implement remedial action.

The subframe concept was developed in response to both of these types of error conditions which could cause error propagation and prevent the display of one or more unusable video frames to the pilot. The video frame is partitioned into horizontal blocks of rows. Each horizontal block is a subframe. The subframe length is command selectable by the pilot to be 20, 10 or 4 lines in length. Each subframe represents in the compressed video data domain a block of independently compressed data. The start of each subframe restarts both the video compression process in space as well as the video reconstruction process on the ground. Each subframe carries with it that data needed to detect its initial point (subframe sync - a 13-bit word), and all compression and overhead parameters required for its reconstruction. Thus, for an uncorrectable error being detected in subframe N, for

example, the subframe error containment function would not permit the propagation of the error into subframe N+1. It should be noted at this point that although each subframe contains compressed video data for a fixed number of video pixels, the number of bits in each subframe varies due to the variable amount of entropy or video information in a given subframe.

Figure 3 shows a conceptual schematic of the means by which subframe replacement operates. The right-most block on the diagram is a digital buffer containing one complete video frame of reconstructed video data - arranged in rows and columns corresponding pixels in the video format. The data shown in the buffer corresponds to frame N. The data shown in the left block corresponds to the reconstructed video data for the succeeding frame. The data for the successor frame N+1 is effectively metered into the digital buffer in discrete subframe parcels. The data for each of these subframes has associated with it a 'line start' and a 'line stop' address. The diagram shows (at the top of the left block) 'error-free reconstructed video'. This represents an integral number of subframes from video frame N+1 which will be read into their respective addresses in the output digital buffer. This overwrites the data from frame N. Lower in the left block is shown 'video data with uncorrectable errors'. This again represents an integral number of subframes. Because the data is faulty, it will not be written into the output buffer. Instead, the data from frame N will be reused for the subframe or subframes associated with the defective data.

SUBFRAME ERROR DETECTION

Two means are provided for the detection of subframe errors. In the first instance, the Reed/Solomon decoder is used to detect faulty Reed/Solomon codewords. This situation automatically gives rise to subframe replacement. If the Reed/Solomon decoder has been able to correct all errors, the data is then reconstruction processed. In the reconstruction processing which is done on a subframe-by-subframe basis, the logic in the reconstructor makes a verification of the pixel count in the subframe. Because of the relatively complex transformations involved in the compression and reconstruction algorithm, it is very unlikely that defective data could be reconstructed incorrectly but have a correct reconstructed pixel count.

The detection of errors by either of the foregoing methods will invoke the subframe replacement process which was described in the previous section.
Figure 4 shows the relationship between the detection of uncorrectable errors by the Reed/Solomon decoder and the compressed subframe data structure. At the top of the diagram is shown the incoming bit stream for a single camera or channel which has already been deinterleaved from the corresponding data for the other channel. Vertical solid markers in the data stream represent Reed/Solomon codeword boundaries after parity and sync symbols have been removed. The vertical dashed lines represent subframe

boundaries. As can be seen in the diagram, these do not have a constant length since the amount of video information is a subframe variable. As noted before, however, the subframes do represent a fixed number of video lines and pixels. Subframe boundaries are defined by a unique (each of the two channels has its own synch word) 13-bit sync word at the start of each subframe. This, in fact, is the means by which the channel deinterleaving occurs.

An “X” is shown in Reed/Solomon codeword M+1 indicating that it contained uncorrectable errors. Codeword M+1, however, contains data from both subframe N and N+1. Since only the trailing end of subframe N is contained in the uncorrectable codeword, it may or may not be affected, depending upon the location of the errors in the codeword. It is, however, viewed as suspect data and, thus, treated as bad data and subframe replacement is again invoked. In this example, a single uncorrectable Reed/Solomon codeword caused subframe replacement for two consecutive frames. On average, a single codeword contains approximately four to five lines of video data. For twenty line subframes, the uncorrectable codeword would typically be internal to a subframe. For ten and four line subframes, the probability is increased that a single codeword would span the boundaries of two subframes.

The Video Compression and Reconstruction System provides for three command selectable subframe sizes: 20, 10 or 4 lines in length. In a heavy RFI environment, the probability of having uncorrectable or residual errors in Reed/Solomon codewords is more likely. Subframe replacement is then more likely. Although testing has shown isolated subframe replacement to be virtually undetectable, subframe replacement does cause portions of the video frame to have older (i.e., 200 msec older) video data in it. In the interests of minimizing the amount of this “older” video data, the smaller subframe sizes may be used. For the 20-line subframe, approximately 8% of the video screen is associated with a subframe. For the 4-line subframe, only 1.6% of the video screen is associated with a subframe. The X in Figure 4 shows the location of an uncorrectable codeword. This necessarily will cause subframe replacement. If a smaller subframe were in use, a smaller portion of the screen would be replaced for the same uncorrectable error.

SIMULATED PERFORMANCE RESULTS

A software RF channel and space communication system simulation exists at NASA-Goddard (GSFC). This simulation software is called CLASS (Reference 4). The CLASS program has been used to conduct some simulation data runs for the OMV video compression system described above. Mr. Robert Godfrey and Dr. Roger Avant of NASA-GSFC and Mr. Ted Kapplan, and Mr. Dave Wampler of STI, Inc., have been instrumental in conducting the programming and computer runs, a formidable task. Figure 5 depicts conceptually those system points at which statistical data is recorded. Table 1 is a

condensed summary of the error statistics for 50-frame runs with 0 db and -2 db EIRP margin at TDRS (relative to no RFI) respectively. A set of 10-frame runs were conducted to determine the approximate EIRP margin (TDRS) at which the video reconstruction would deteriorate to an unusable picture. The runs were made for 0 db, -1 db and -2 db. A comparison of the error rate out of the Viterbi decoder for the 10- and 50-frame runs is of interest because it can indicate whether 50-frames will be sufficient to prove a statistical sample. These results are indicated in Table 2. (A simulation run of 50-frames takes approximately 12 hours.)

The results of Table 2 indicate that the error statistics after the Viterbi decoder are roughly the same for the 10-frame and 50-frame runs. However, there are no available statistics on the random/burst mixture, so we cannot make any judgement on 50-frame statistics for random/burst mixture. A 200-frame run is planned to compare with the 50-frame runs. Also from Table 2, it is apparent that the video link deteriorates to unusable between -1 db and -2 db EIRP margin (TDRS). Additional 50-frame runs are planned at a EIRP margin (TDRS), probably around -1.0 db EIRP margin (TDRS), that provides some frames with replacements and some frames with no replacements.

Table 1 illustrates the system performance for a good working video link (0 db) and a completely unusable video link (-2 db). In fact, at -2 db EIRP margin (TDRS), there were no video frames reconstructed. The following discussion will explain why the results in Table 1 for both the 0 db case and the -2 db case are as one would expect.

First, one must realize that deinterleavers of any type will disburse a long burst of errors into smaller bursts distributed more or less on a periodic basis throughout the deinterleaved symbol stream. Also, a uniformly random distribution of error events into a deinterleaver produces a uniformly random distribution of errors in the output symbol stream and a uniformly random distribution of short bursts input will create an approximate uniform random distribution of small bursts in the output symbol stream. With this in mind, inspect the data in Table 1.

The statistics into the periodic convolutional deinterleaver (DPCI) are seen to be a random mixture of random and burst error events. The mean length, $\bar{L}$, of the bursts for both 0 db and -2 db runs are fairly short, of the order of 11 to 18 symbols long. Even within two standard deviations, the burst error lengths are only of the order of 26 and 44 symbols. After the DPCI, the error events are still seen to be a rather random mixture of random and burst error events with bursts of mean lengths 12 and 18 symbols and, within two standard deviations, lengths of 30 and 48 symbols, respectively.

The Viterbi decoder with a free distance of 10 across the constraint length of about 64 symbols can correct most error bursts of 4 to 5 symbols in a span of about 64 symbols.

The output of the DPCI for the 0 db runs produces a burst event on the average of 1.6 burst events every 100 symbols. These bursts have an average length of 12 symbols and an average of 3 errors per burst. Thus, the Viterbi should be able to correct most of these error events; and it does a good job, reducing the cumulative output bit error rate from $6.478x10^{-2}$ to $3.335xl0^{-4}$ (a factor of 200 reduction). However, at -2 db EIRP margin (TDRS), the burst error events out of the DPCI occur at a rate of 2.2 per 100 symbols with an average length of 18 symbols and an average of 4 to 5 errors per burst. Now the Viterbi decoder should experience difficulty decoding the average burst, especially with an occasional random error included in the 64 symbol constraint length; and it does have trouble, reducing the cumulative output bit error rate from $11x10^{-2}$ to only $2.4xl0^{-2}$ (a factor of 5 reduction).

The error statistics out of the Viterbi indicate a fairly uniform mixture of random and burst error events and after the R/S deinterleaver, a similar error makeup is expected (there are no long strings of bursts to deinterleave).

As a result, the probability of symbol errors in the 255-symbol (2040 bit) R/S codeword is conservatively approximated by assuming all error events are simply random events. Thus, a cumulative error event rate of $14.225x10^{-5}$ ($6 . 74x10^{-5}$ plus $7.48x10^{-5}$) (0 db margin case) will produce an average of .29 errors per 2040 bits; hence, the R/S decoder should correct these, and it does. No subframe replacements occurred, and no R/S codewords failed to decode after initial synchronization was achieved. However, for the -2 db margin runs, the Viterbi output bit error event rate of $5.668x10^{-3}$ ($1.879x10^{-3}$ plus $3.789x10^{-3}$) produces an average of 11.56 error events per 2040 bits or an average of 11.56 errored R/S symbols per R/S codeword. Now the R/S decoder should fail to decode the errors, and it does. (All video frames failed to reconstruct.)

An interesting observation from the Table 1 test results is that a 0 EIRP margin (TDRS) with no RFI should produce 10^{-5} error rate after the Viterbi decoder. In the 10- and 50-frame test, high RFI runs, the Viterbi decoder error rate at 0 EIRP margin (TDRS) was approximately $3x10^{-4}$. Thus, the RFI induced on the channel raises the average error rate by a factor of approximately thirty (30).

An estimate of initial system synchronization time was made (Reference 5). An estimate for the average Viterbi search and best subframe replacement synchronization case was that 140 ms would be the acquisition time. The test runs for 50 frames took 30 R/S codewords to acquire synchronization. This is 68,040 bits. At a bit rate of 486,000 bits per second, 140 ms is 68,040 bits. What a lucky coincidence! In fact though, the first two video frames (approximately 200,000 bits for each frame) are lost due to the need for the VRU to have a correct frame header for each frame to achieve synchronization and the fact that channel frames are interleaved.

Another interesting observation is that after initial synchronization, neither bit sync, Viterbi sync, nor R/S sync were dropped in the -2 db runs. Although "hits" were occurring in the R/S sync words, there was never a danger of losing R/S sync; the video was lost because, there were simply too many channel errors to correct.

REFERENCES

1. A. Netravali and B. Haskell, Digital Pictures, 1988 AT and T Bell Laboratories, Plenum Press, page 305.
2. R. C. Gonzalez and P. Wintz, Digital Image Processing, 2nd Edition, 1987, Addison and Wesley, ISBN 0-201-11026-1, pages 293-300.
3. A. Netravali and B. Haskell, Digital Pictures, 1988 AT and T Bell Laboratories, Plenum Press, ISBN 0-306-42791-5, pages 379-380.
4. McKenzie T.M. and H. Choi, "User's Guide to CLASS's Analytical Computer Program for Bit Error Rate With RFI," Revision 3, TR-8512-23, LinCom, L.A., CA, Dec. 1985.
5. Ingels F., "Analysis of Space Telescope Data Collection System," NAS8-36044, Quarterly Report, May 1989.

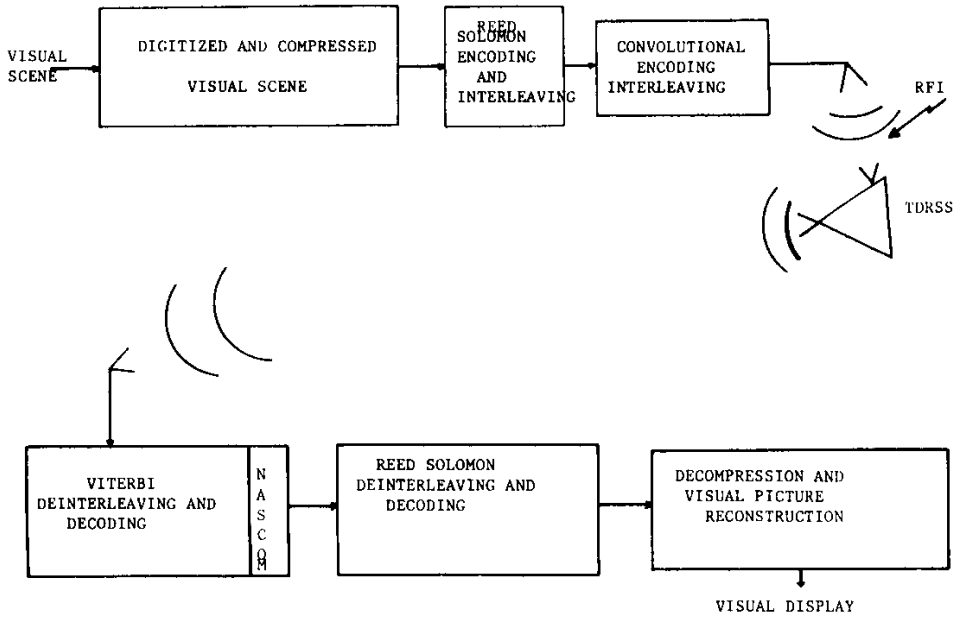


FIGURE 1. COMPRESSED VIDEO RETURN LINK CODING TECHNIQUES

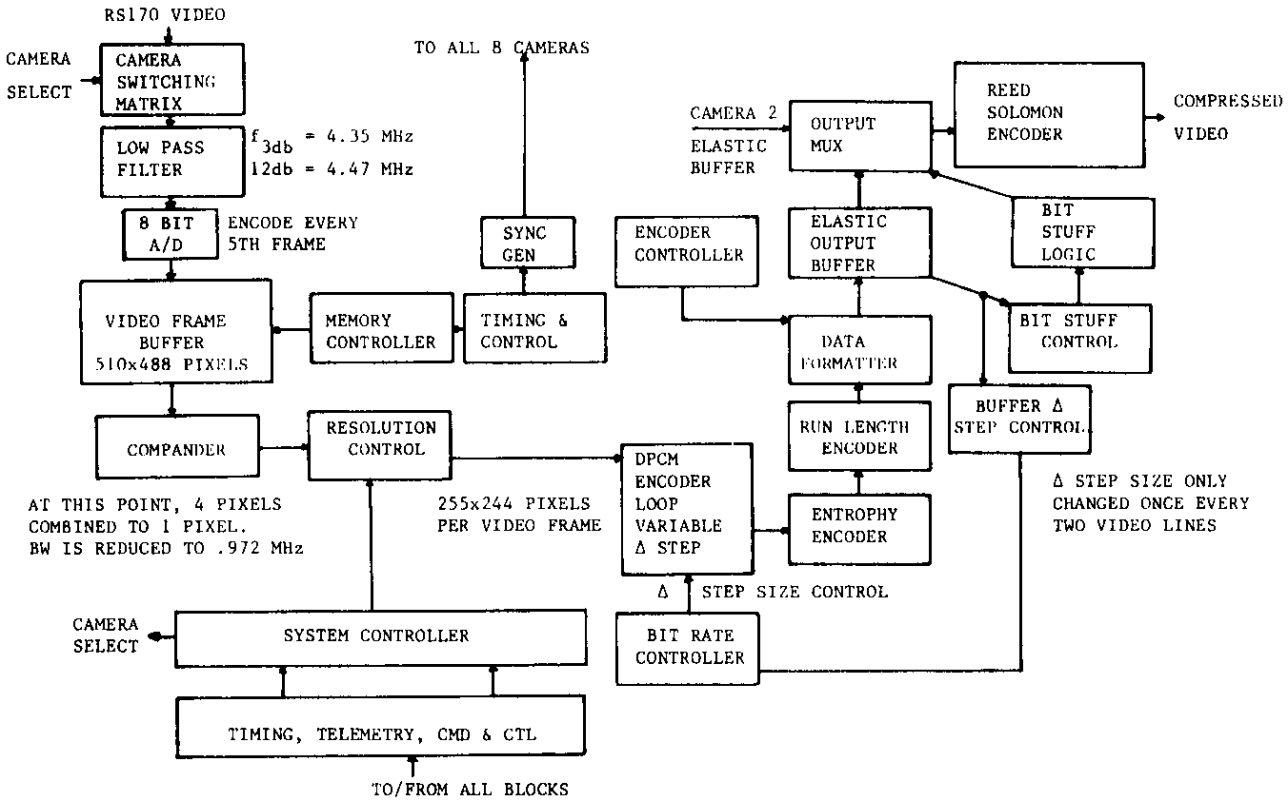


FIGURE 2 VCU BLOCK DIAGRAM

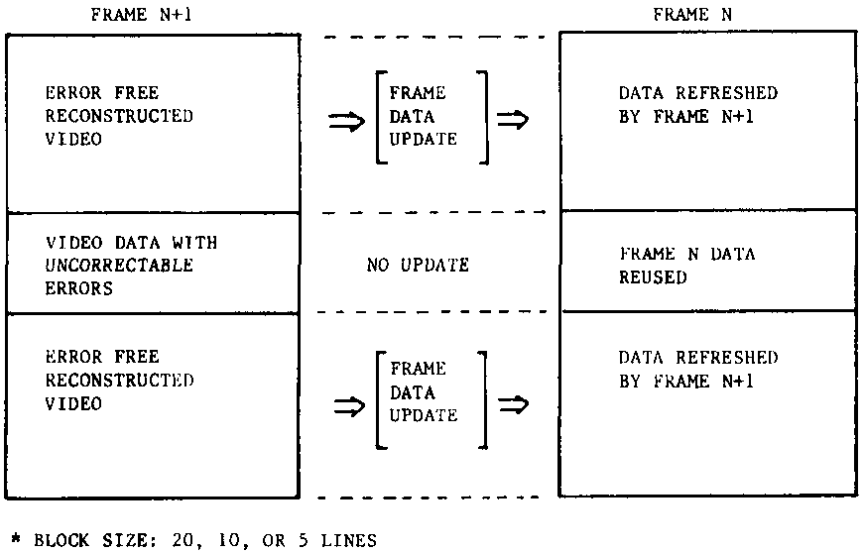


FIGURE 3 VIDEO FRAME OUTPUT BUFFER

TABLE 1

STATISTICS FOR TWO 50 FRAME RUNS 17 APRIL 1989

STATISTIC	BEFORE DPCI (II DATA)		AFTER DPCI (III DATA)		VITERBI DECODER OUTPUT	
EIRP MARGIN	0 db	-2 db	0 db	-2 db	0 db	-2 db
1. Bit Slip Rate	-0-	-0-	-0-	-0-	-0-	-0-
2. Number of Random Errors	112,390	86,646	135,688	76,320		
3. Random Error Rate	1.229×10^{-2}	9.447×10^{-3}	1.484×10^{-2}	8.35×10^{-3}	6.74×10^{-5}	1.879×10^{-3}
4. Number of Burst Errors	126,137	192,000	147,850	202,471		
5. Burst Error Rate	1.379×10^{-2}	2.1×10^{-2}	1.617×10^{-2}	2.21×10^{-2}	7.48×10^{-5}	3.789×10^{-3}
6. Total Number of Errors	592,311	1,010,088	592,104	1,009,729		
7. Cumulative Error Rate	6.478×10^{-2}	.1105	6.478×10^{-2}	.1105	3.335×10^{-4}	2.461×10^{-2}
8. Average Error Burst Length, $\bar{L}$	10.694	16.66	12.109	18.514	5.657	10.28
9. Std. Dev. of $\bar{L}$	8.0824	14.16	9.206	15.983	3.837	9.64
10. Average Errors Per Burst, $\bar{E}/\bar{B}$	3.8047	4.809	3.087	4.61	3.555	5.998
11. Std. Dev. of $\bar{E}/\bar{B}$	2.173	3.342	1.502	3.07	1.938	5.366
12. Average Error Spacing Per Burst, $\bar{ES}/\bar{B}$	2.921	3.39	4.316	3.856		
13. Std. Dev. of $\bar{ES}/\bar{B}$	3.051	3.06	3.116	3.034		
14. Viterbi In-Sync Error Rate					3.335×10^{-4}	2.46×10^{-2}
15. Number of Bits for Viterbi to Reacquire Sync					No Sync Drop	
16. Number of Video Frames With Subframe Replacements					None	No Video Frames Reconstructed

TABLE 2

COMPARISON OF 10 FRAME AND 50 FRAME RUNS
(CONFIGURATION 1)

No. of Frames/Run	EIRP*: 0 db	-1 db	-2 db
Subframe Replacements:			
10 Frame Run	0	0	ALL
50 Frame Run	0	-	ALL
Bit Error Rate After Viterbi Decoder:			
10 Frame Run	3.09×10^{-4}	2.43×10^{-3}	2.33×10^{-2}
50 Frame Run	3.34×10^{-4}		2.46×10^{-2}
Bit Error Rate After Bit Sync and Matched Filter But Before DPCI:			
50 Frame Run	6.478×10^{-2}		11.04×10^{-2}

25 frames per camera channel, 50 frames total, statistics based on 50 frames. 20 lines per subframe, 12 subframes per frame, approximately 20 R/S codewords per frame.

* EIRP margin at TDRS for no RFI. (The high RFI channel model was used for the runs.)

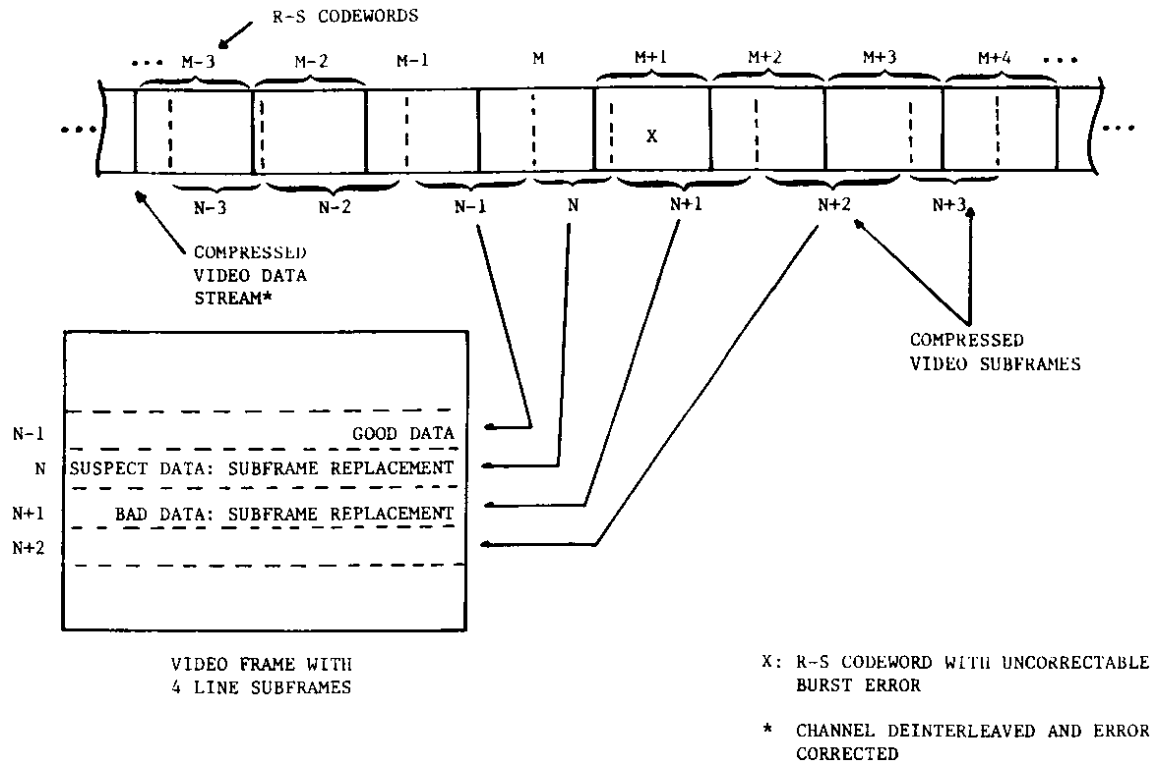
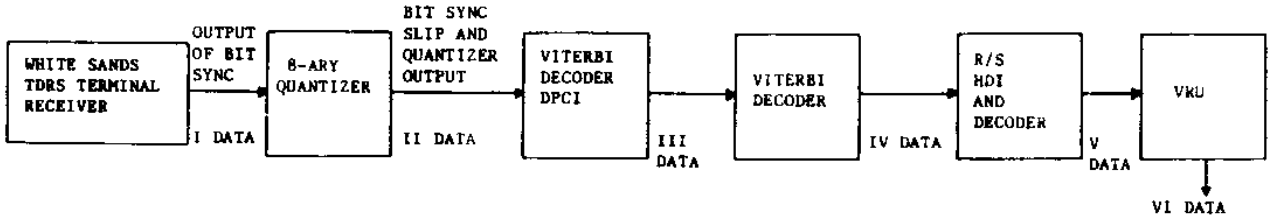


FIGURE 4 R-S CODEWORD, VIDEO SUBFRAME RELATIONSHIP



DATA POINT	CONCISE SUMMARY
I DATA	BIT SYNC OUTPUT CLOCK JITTER SPECTRUM AND SYMBOL (BIT SYNC CLOCK OUTPUT) TIME INTERVAL VARIATION STATISTICS
II DATA	BIT SYNC BIT SLIP STATISTICS, 8-ARY TRANSITION PROBABILITIES, RANDOM AND BURST ERROR STATISTICS PRIOR TO DPCI
III DATA	BURST AND RANDOM ERROR STATISTICS AFTER DPCI, BUT BEFORE VITERBI DECODING
IV DATA	VITERBI DECODED OUTPUT BURST ERROR STATISTICS AND SYNC STATISTICS. NUMBER OF BITS IT TAKES FOR VITERBI/DPCI SYSTEM TO RECOGNIZE SYNC LOSS AND NUMBER OF BITS IT TAKES TO REACQUIRE SYNC
V DATA	R/S DECODER SYNC STATISTICS AND UNDECODABLE, DECODABLE AND MISCORRECTED R/S WORD STATISTICS
VI DATA	TOTAL CHANNEL BIT ERROR STATISTICS