Video Compression Using Conditional Replenishment and Motion Prediction

DAVID N. HEIN, MEMBER, IEEE, AND NASIR AHMED, SENIOR MEMBER, IEEE

Abstract—**A study of a low-rate monochrome video compression system is presented in this paper. This system is a conditional-replenishment coder that uses two-dimensional Walsh-transform coding within each video frame. The conditional-replenishment algorithm works by transmitting only the portions of an image that are changing in time. This system is augmented with a motion-prediction algorithm that measures spatial displacement parameters from frame to frame, and codes the data using these parameters. A comparison is made between the conditional-replenishment system with, and without, the motion-prediction algorithm. Subsampling in time is used to maintain the data rate at a fixed value. Average bit rates of 1 bit/picture element (pel) to 1/16 bit/pel are considered. The resultant performance of the compression simulations is presented in terms of the average frame rates produced.**

Key Words—**Video compression, conditional replenishment, motion prediction, Walsh transforms.**

Index Code—**P7j, P3k.**

I. INTRODUCTION

THE CONDITIONAL-REPLENISHMENT technique, as reported in this paper, is a step toward achieving large compression gains with minimal quality reduction. Conditional replenishment, along with intraframe compression techniques, allows us to vary the effective sampling rates at different points within an image to obtain an overall lower data rate. Basically, this is done by transmitting only the changing portions of an image and having the decoder retain the static part of the image. This technique has the potential of producing very low data rates for images with small amounts of motion. The amount of compression that can be achieved through this technique depends to a large extent on the rate of change within the image, and typically varies from 10 to 1, to less than 2 to 1. An additional 3 to 1 reduction in rate is obtained in the compression simulations by the intraframe coding of data blocks using a two-dimensional variable-rate Walsh-Hadamard Transform (WHT) coder.

Since a conditional replenishment system must transmit, in its entirety, the data in the changing portions of an image, even greater gains could be obtained by modeling the changes and sending information about the types of changes that are taking place. The motion-prediction algorithm attacks this problem by assuming all changes are in the form of translational motion. It measures the displacement of a portion of an image from one frame to the next. If it has been determined that the change is really of this type, it follows that only the vertical and horizontal displacements need be transmitted. Otherwise, the changed data are transmitted as before. An additional 33-percent rate reduction is achieved by using motion prediction. Computer simulations have demonstrated that data rates of 2 or 4 Mbit/s can be achieved, while still retaining good fidelity in the image.

The results reported in this paper are very promising. They show that a video-compression system that utilizes intraframe conditional-replenishment and motion-prediction techniques is capable of about 8 or 16 to 1 data compression.

II. BACKGROUND

Kell [1] was probably the first to propose a conditional-replenishment system for data-rate compression of video images. The system, for which he obtained a patent in 1929, operates by sending the difference between two successive frames, rather than the frame itself. However, a practical hardware implementation of this system has not been possible until recently.

During the 1960's, Seyler [2], [3] did much of the basic work in measurement and interpretation of the statistical nature of video differences between frames. He considered the coding difficulties that occur in a simple frame-difference coder when a large amount of motion is present, or when scene changes occur. For large amounts of motion, the coder must use alternate coding methods to prevent the rate-buffer memory from overflowing. One solution, presented by Seyler, is to reduce the spatial resolution of the image that is transmitted during periods of high motion.

In 1969, Mounts [4] reported the first real-time conditional-replenishment system. This system took the differences between successive frames and detected changing areas by thresholding the differences. The value of the threshold depended on how full the rate buffer was. When the buffer was empty, the threshold was set to zero and all nonzero differences were detected as moving areas. If the rate buffer was almost full, the threshold was set to a very large value so that only the most extreme motion was detected. If the buffer became full, all coding was stopped until the buffer had emptied sufficiently. This resulted in a displeasing breakdown in the image. Connor, Candy, and others [5], [6] developed methods to eliminate the picture breakdown by subsampling the image both temporally and spatially. This technique gained a significant amount of compression with little loss in subjective quality.

Rocca and Zanoletti [7] investigated the data compression obtainable by performing movement compensation before doing interframe differential coding. They used computer-

Manuscript received March 15, 1983; revised March 20, 1984. This work was supported in part by the NASA-Ames Research Center, Moffett Field, CA.

D. N. Hein is with Compression Laboratory, Inc., San Jose, CA 95131.

N. Ahmed is with the Electrical and Computer Engineering Department, University of New Mexico, Albuquerque, NM 87131.

0018-9375/84/0800-0134$01.00

generated data based on a statistical model of moving imagery. Later, Brofferio and Rocca [8], [9] employed this method on actual video data and obtained excellent results. In this research, the algorithm distinguished the various objects within the image, and would measure the frame-to-frame displacements for each object. These displacements were utilized to modify the contents of a frame memory containing the previous frame, to compensate for the motion in the image. The modified data were subsequently used in an interframe differential pulse-code modulation (DPCM) coder. A significant reduction in the entropy of the interframe differential signal was obtained through the use of motion compensation. They measured reductions of entropy from about 33 to 50 percent.

Haskell [10], [11] measured the effectiveness of velocity-adaptive linear interframe predictors for DPCM coding. He subdivided video-telephone images into small subpictures and measured a velocity, or frame-to-frame spatial displacement, for each block. This velocity was subsequently used to design an optimal predictor for each subpicture. The weighting coefficients for the predictor was adjusted to compensate for the displacement of the block from one frame to the next. Haskell found that a considerable reduction in the entropy of the frame differences was obtained when using small block sizes. However, even a block size of one field gave significant entropy reduction. He also obtained about a 33-percent reduction in rate using motion prediction.

Recently, Netravali, Stuller, and Robbins [12]-[14] have been studying motion-compensation systems using iterative methods of computing the displacements from frame to frame. As the frame is being coded, the displacement from the previous areas are used to predict the displacement for the next area that is to be coded. This has been combined with a gradient-type operation to allow relatively precise measurements to be made. They have concentrated primarily on using this method with DPCM coders. However, they have also studied the use of this method in transform compressors, and have studied motion compensation in both the data and transform domains.

In 1977, Jones [15], [16] reported experiments with conditional-replenishment systems that utilized transform techniques for coding the image data. These systems employed a fixed grid of subpictures for detecting changes and addressing the changed portions of an image. Thus instead of assigning changes to single picture elements and grouping them together for transmission, changes were detected on an 8 X 8 block of picture elements and a single bit was transmitted for each block to signify a changed or unchanged status. This system is quite efficient in addressing the changing portions of an image and is also consistent with the two-dimensional transform coding techniques.

The main objective of this paper is to present the results of a study which consider two basic modifications of the conditional-replenishment system proposed by Jones [15], [16]. These are: 1) inclusion of a variable-rate code which was designed to handle the WHT coefficients of 8 X 8 subpictures, and 2) development and inclusion of a motion prediction algorithm. Simulation results are included to demonstrate that these modifications result in an appreciable improvement in the performance of the video compression system.

III. ALGORITHM DESCRIPTION

Block diagrams of the encoder and decoder for the basic compression system simulated in this study are shown in Figs. 1 and 2, respectively. In the encoder, a monochrome video signal is sampled and digitized, and fed into a memory that reformats the video data into 8 X 8 blocks. Each subpicture then passes through a two-dimensional WHT stage.

The transformed data is stored in a frame memory for use one frame-time later. Simultaneously, each block is classified by the mode detector. The mode designation will subsequently be used by the variable-rate coder when coding a block and computing the required rate, if the block needs to be transmitted. The difference between the data from the blocking memory and the data from the previous frame is computed by the change detector. The data from the previous frame are obtained from a frame memory in the encoder that tracks the image being displayed at the decoder. This information is also fed to the motion predictor which measures the spatial displacement of a subpicture from the previous frame.

When the total difference between the two blocks is greater than a fixed threshold, the block is designated as changed. A "1" bit is put into the corresponding position in the change map to indicate that a change has occurred. The change map consists of one bit for each 8 X 8 subpicture. The first field of a changed-block pair is transmitted using the variable-rate WHT coder. The decoder copies the first field into the second when displaying a replenished subpicture. This technique is acceptable because the eye cannot see with as much spatial resolution when an image is changing as it can without change.

The algorithm employed in this study for coding changed blocks is shown in Fig. 3. A subpicture is checked by the change detector to determine whether it requires replenishment or not. If the difference exceeds the change threshold, then the subpicture needs to be replenished. The subpicture is next checked to see whether it can be transmitted by displacing the data from the previous frame. This check is accomplished by comparing the block to be coded, with a 22 X 22 window centered around the corresponding block in the decoder tracking memory. The subpicture is shifted to all possible positions within the window and a difference is computed for each possible displacement. The displacement with the minimum difference determines the displacement of the block. This is depicted in Fig. 4.

The minimum difference is compared to the change threshold and if the difference is less than the threshold, the block is coded by sending only the horizontal and vertical displacements. Otherwise, the block is replenished by sending the statistically compressed data.

Frame repeat is used when large amounts of motion occur. If there are enough changed blocks within a frame that it takes more than one frame time to transmit all of the data, the decoder continues to display the same frame until all the changed blocks have been sent. Any excess rate is used by the encoder to refresh blocks that have not changed re-

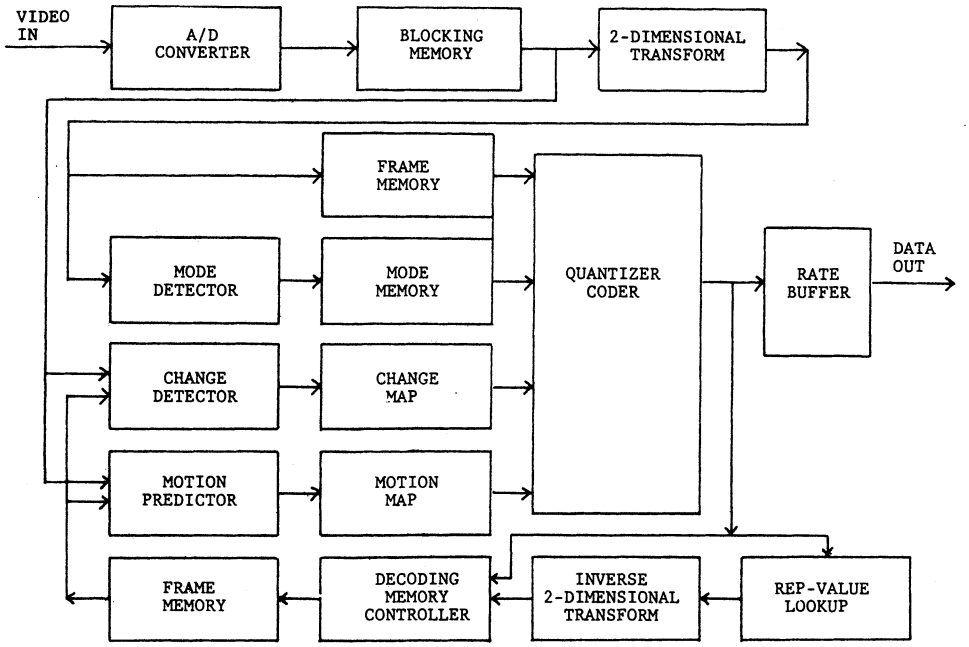

Fig. 1. Conditional replenishment encoder.

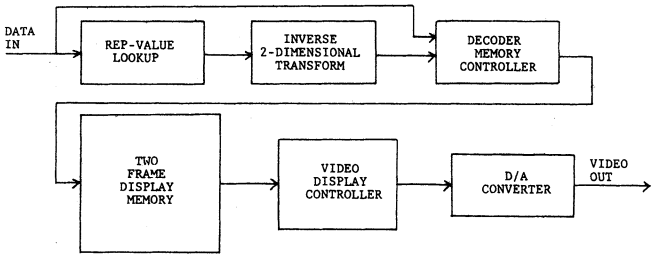

Fig. 2. Conditional replenishment decoder.

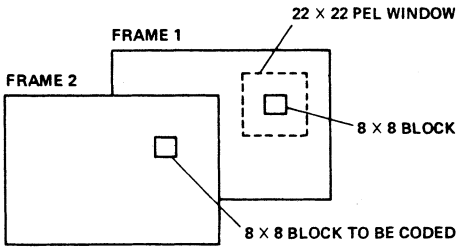

Fig. 4. The motion-prediction algorithm.

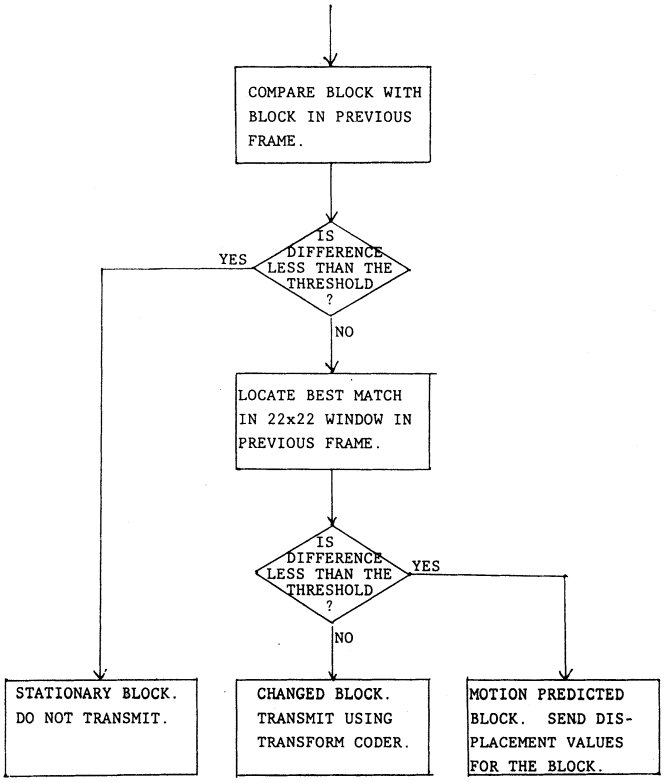

Fig. 3. Motion prediction flow diagram.

cently. These blocks are also coded using the variable rate WHT coder. However, in this case the data for both fields are sent. The encoder decides which of the static blocks to refresh by keeping track of their ages from when they were last refreshed, or replenished, and refreshing the oldest of the static blocks.

IV. THE TRANSFORM CODER

The interframe coder used in this study is a variable rate WHT coder. The basis functions for the 8 X 8 WHT are shown in Fig. 5, where [17], [19]

$$A_{km} = \text{wal}\,(k, x)\ \text{wal}\,(m, y)$$

where wal (k, x) and wal (m, y) denote continuous Walsh functions with indexes k and m, and x and y are spatial variables. The sequency corresponding to a given index i is $i/2$ or $(1 + i)/2$ for i even and odd, respectively. After performing the transform, the coder classifies the subpicture into one of six possible modes [18], [19]. Each mode requires a different amount of rate to code the 8 X 8 subpicture. This classification is done by comparing the transform coefficient values to each of five sets of thresholds. The subpicture is assigned to the first mode for which all the transform coefficients are below their respective thresholds. If the

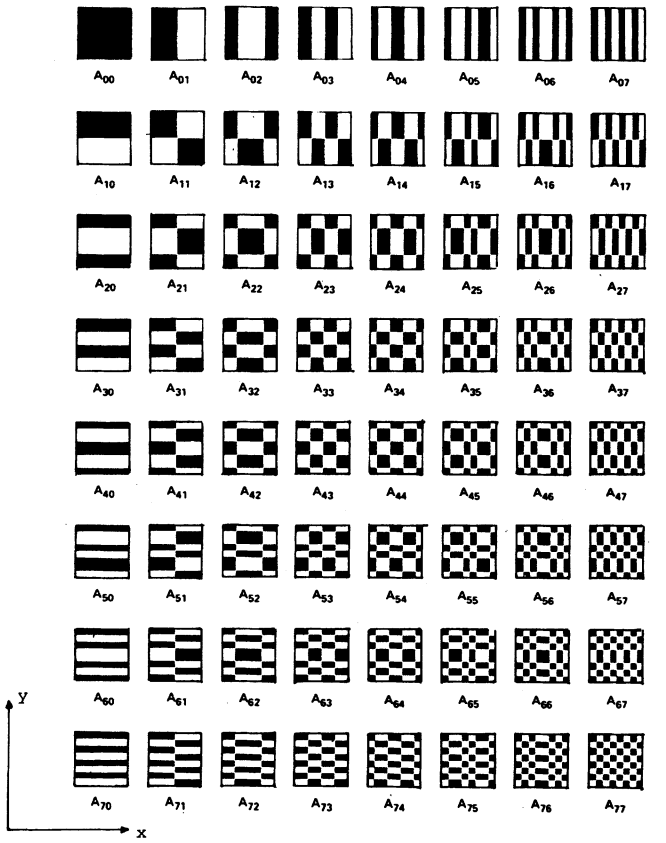

Fig. 5. Basic functions for the 8 WHT.

transform coefficients exceed all the threshold sets, the block is assigned to a sixth mode which does no compression on the data. The threshold sets used in this study are shown in Table I. Dashes indicate that no test is made on the coefficient for the threshold set.

The number of bits assigned to each coefficient for each mode is shown in Table II. The average rates required for each mode can be determined by summing together the individual number of bits assigned to each coefficient and dividing by 64, which is the number of samples per block. The bit assignments were determined by observing the value of the threshold for a certain coefficient and assigning the necessary rate to code that coefficient with full accuracy over the range indicated by the thresholds. For mode six, all of the coefficients are assigned the full 8-bit range. Although the rate produced by the intraframe coder is quite variable, from 0.86 to 8.00 bits/picture element (pel), the rate for each block is buffered over a frame time to produce a continuous rate. If the number of changed blocks is small enough, the overall rate will be less than that allocated for a frame. Otherwise, more than one frame time will be required to transmit the changed data.

Table III below gives the average number of bits per pel and the total number of bit allocated for each of the 6 modes used. The lowest mode requires a very low rate of 0.86 bit/pel, and the rate increases for higher modes up to 8 bit/pel, for the highest mode. In the 8 bit/pel mode, no data compression is being done. The modes are determined by the ranges

TABLE I
THRESHOLDS USED IN MODE DETECTION

Threshold set 1

-	8	4	4	2	2	2	2
8	4	2	2	1	1	1	1
4	2	1	1	-	-	-	-
4	2	1	1	-	-	-	-
2	1	-	-	-	-	-	-
2	1	-	-	-	-	-	-
2	1	-	-	-	-	-	-

Threshold set 2

-	16	8	8	4	4	4	4
16	8	4	4	2	2	2	2
8	4	2	2	1	1	1	1
8	4	2	2	1	1	1	1
4	2	1	1	-	-	-	-
4	2	1	1	-	-	-	-
4	2	1	1	-	-	-	-

Threshold set 3

-	32	16	16	8	8	8	8
32	16	8	8	4	4	4	4
16	8	4	4	2	2	2	2
16	8	4	4	2	2	2	2
8	4	2	2	1	1	1	1
8	4	2	2	1	1	1	1
8	4	2	2	1	1	1	1
8	4	2	2	1	1	1	1

Threshold set 4

-	64	32	32	16	16	16	16
64	32	16	16	8	8	8	8
32	16	8	8	4	4	4	4
32	16	8	8	4	4	4	4
16	8	4	4	2	2	2	2
16	8	4	4	2	2	2	2
16	8	4	4	2	2	2	2
16	8	4	4	2	2	2	2

Threshold set 5

-	-	64	64	32	32	32	32
-	64	32	32	16	16	16	16
64	32	16	16	8	8	8	8
64	32	16	16	8	8	8	8
32	16	8	8	4	4	4	4
32	16	8	8	4	4	4	4
32	16	8	8	4	4	4	4
32	16	8	8	4	4	4	4

TABLE II
VECTOR COEFFICIENT BIT ASSIGNMENTS

Mode 1

8	4	3	3	2	2	2	2
4	3	2	2	-	-	-	-
3	2	-	-	-	-	-	-
3	2	-	-	-	-	-	-
2	-	-	-	-	-	-	-
2	-	-	-	-	-	-	-
2	-	-	-	-	-	-	-
2	-	-	-	-	-	-	-

Mode 2

8	5	4	4	3	3	3	3
5	4	3	3	2	2	2	2
4	3	2	2	-	-	-	-
4	3	2	2	-	-	-	-
3	2	-	-	-	-	-	-
3	2	-	-	-	-	-	-
3	2	-	-	-	-	-	-
3	2	-	-	-	-	-	-

Mode 3

8	6	5	5	4	4	4	4
6	5	4	4	3	3	3	3
5	4	3	3	2	2	2	2
5	4	3	3	2	2	2	2
4	3	2	2	-	-	-	-
4	3	2	2	-	-	-	-
4	3	2	2	-	-	-	-
4	3	2	2	-	-	-	-

Mode 4

8	7	6	6	5	5	5	5
7	6	5	5	4	4	4	4
6	5	4	4	3	3	3	3
6	5	4	4	3	3	3	3
5	4	3	3	2	2	2	2
5	4	3	3	2	2	2	2
5	4	3	3	2	2	2	2
5	4	3	3	2	2	2	2

Mode 5

8	8	7	7	6	6	6	6
8	7	6	6	5	5	5	5
7	6	5	5	4	4	4	4
7	6	5	5	4	4	4	4
6	5	4	4	3	3	3	3
6	5	4	4	3	3	3	3
6	5	4	4	3	3	3	3
6	5	4	4	3	3	3	3

Mode 6

8	8	8	8	8	8	8	8
8	8	8	8	8	8	8	8
8	8	8	8	8	8	8	8
8	8	8	8	8	8	8	8
8	8	8	8	8	8	8	8
8	8	8	8	8	8	8	8
8	8	8	8	8	8	8	8
8	8	8	8	8	8	8	8

TABLE III
AVERAGE BIT RATE FOR EACH MODE

MODE	TOTAL NUMBER OF BITS	AVERAGE BIT RATE
1	55	0.86
2	98	1.53
3	163	2.55
4	240	3.75
5	303	4.73
6	512	8.00

of the coefficients for each data block. Blocks with very small values for the transform coefficients will be classified as mode 1 blocks, and will require only 0.86 bit/pel to code. Mode 1 blocks contain very little detail, and appear in areas of almost constant intensity.

On the other hand, blocks with a great amount of detail will require more bits to represent them. The range of their transform coefficients will be greater and this will cause them to be classified in a higher mode. These types of blocks typically appear along the edges in an image. Graphics-type images, such as plots and text information, will frequently contain a large number of edges and will require a higher rate.

V. THE CHANGE DETECTOR

The change detector operates by taking the difference between an 8 X 8 subpicture and the corresponding 8 X 8 subpicture in the previous frame. The differences are squared and summed together to find the total squared difference between the two blocks, i.e.,

$$D = \sum_{i=1}^{N} (V_i - W_i)^2 \tag{1}$$

where D is the difference, V_i is the ith data component of the block to be transmitted, and W_i is the ith data component of the corresponding block in the previous frame.

The difference value could also be computed in the transform domain by taking the squared difference of the WHT coefficients. It can be shown that this is true by referring to Parseval's theorem, which states that the squared difference computed in the data domain is equivalent to that computed in the transform domain for orthonormal transforms. This value, which is computed by the change detector, is compared to a fixed threshold to determine whether there is a change or not. The change threshold was experimentally determined by computing and plotting histograms of the experimental differences measured from the video data, as shown in Fig. 6. The curve in Fig. 6 represents the percent number of subpictures as a function of the frame-to-frame difference, or mean squared error (MSE). This curve was obtained from the sequence of images used in this study.

The values in the histogram from 0 to 3.0 represent the random noise present in the image. This peaks at a value of about 1.0. The region above 3.0 represents actual change occurring in the image. The change threshold was set to 3.0 throughout the course of this study. This value for the change threshold is high enough to be above the noise, but low enough to detect most of the real changes. Because of the high noise factor, there are some changes that occur below the threshold and hence are not detected. Undetected changes usually happen in areas where there is very little contrast (i.e., a small dynamic range). Another problem exists in blocks that contain large edges or a wide range in pixel values. Here a very small shift in the position of the pels will result in a very large value for the squared difference. This value may exceed the threshold, even though the change itself may not be visible.

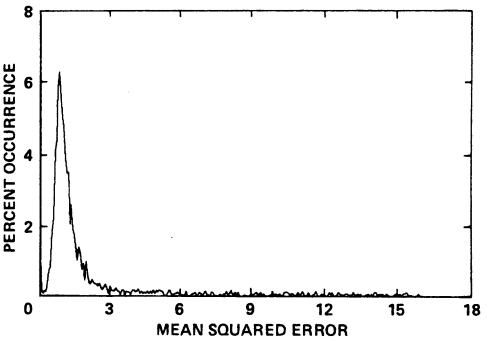

Fig. 6. Distribution of frame-to-frame differences.

VI. MOTION PREDICTION

The algorithm employed in this study for performing motion prediction operates in the following manner. A subpicture is checked by the change detector to determine whether it requires replenishment or not. If the subpicture needs to be replenished, it is next checked to see whether it can be transmitted by displacing the data from the previous frame. This is accomplished by comparing the block to be coded with the corresponding block in the decoder memory and the eight immediately neighboring blocks, as shown in Fig. 4. The subpicture is shifted to all possible positions within a 22 X 22 window centered around itself, and a difference is computed for each possible displacement. This difference is computed by summing the squares of the differences for each pel in an 8 X 8 block as described for the normal conditional replenishment algorithm. The differences are computed in the data domain, rather than in the transform domain, because of the difficulty of computing transform coefficients for spatially shifted data. The displacement that provides the minimum difference is determined, and this is used as the displacement of the block.

The resulting minimum difference is used to determine if the motion prediction was successful. This difference is compared to the same change threshold that is used for the conditional-replenishment determination. If this difference is greater than the change threshold, the block is coded using the transform coding technique described previously. However, if the difference is less than the change threshold, the block is coded by sending the displacement. The displacements are limited in range from -7 to $+7$ in both the vertical and horizontal directions and, therefore, only 8 bits are required to transmit them to the decoder. The decoder will update its display by shifting the data it has stored from the previous frame by the transmitted displacement factors.

As before, in the transform replenishment, the block in the second field is copied from that in the first field. Thus only 8 bits are used to update the data in an 8 X 8 block for both the odd and even fields of a video frame. The average number of bits per pel required is only $8/(2 \times 64) = 1/16$ bit/pel. On the other hand, the average rate required to code the conditionally replenished blocks using the WHT coder, discussed earlier, was about 1.5 bit/pel. This is a factor of 24 times the rate required for motion prediction. Using this technique, images could be transmitted at extremely low

rates if all the changed subpictures could be coded by motion prediction. Unfortunately, this does not happen for actual images. However, it will be shown that a significant percentage of the subpictures can be transmitted in this manner.

VII. RATE COMPUTATION

As was previously noted, the nonstationarity in the number of changes is buffered out by varying the number of frame repeats at the decoder. Other methods of smoothing out the data rates are to vary the value of the change threshold, or the resolution at which the data are sent. However, these methods were not used in the simulations represented in this paper.

The conditional-replenishment system processes the video data in two passes. During the first pass, the system determines the rate required to code each block, whether each block has changed or not, whether a block can be sent using motion prediction, and the total rate needed for transmitting the changed subpictures. During the first pass, each subpicture is also coded and stored in a frame memory. At the beginning of the second pass, the coder determines if any frame repeats are necessary, and the number required. Once the encoder has determined which blocks are to be sent and how they are to be coded, they are transmitted on the second pass.

Frame repeating is a method of reducing the data rate by subsampling in time, and reducing the effective frame rate from 30 frames/s. The encoder does this by skipping over frames, sending the data for only some of the frames, and eliminating the rest. The decoder will display the frames that are transmitted and repeat them at the refresh rate of 30 frames/s until the next frame is received. Since excessive frame repeating will cause a jerky motion, this technique is used only when the encoder requires a higher rate to send a frame than is available during one frame time.

The total rate necessary to send the changed data is computed by adding all the rates required for the changed blocks. The rate required to code a block consists of the sum of the individual bit allocations for each coefficient plus three bits of overhead to indicate in which mode the block is being coded. This is added to the rate needed to transmit the change map to obtain the total transmission rate required for replenishing the changed blocks. If the number of blocks that have changed is small, it is possible for this rate to be less than the total number of bits available during one frame time. When this occurs, the encoder will have more rate at its disposal than is required to send the image. In the computer simulations, this extra rate was used to refresh the nonchanging portion of the image, and thus assist in clearing up any transmission errors that may have occurred in the channel.

Quite often a large portion of the subpictures will be changing in an image. This may cause the transmission rate required to send a frame to be larger than the number of bits available during one frame time. When this happens, the encoder must signal the decoder to continue to display the last frame it received until the next frame has been completely transmitted. For a very active image, it may take several frame times to transmit one frame. In general, the number of repeats necessary is given by:

$$\text{REPEAT} = \text{INT}\,[(\text{TOTAL} + \text{REMIN})/\text{RATE}] \tag{2}$$

where TOTAL is the total rate needed to send the changed blocks, REMIN is the minimum rate reserved for refresh, RATE is the rate available during one frame time, and INT indicates that the integer value of the quotient is to be used. At the decoder, each frame is displayed once, plus the number of repeats given by REPEAT.

The total rate remaining for refreshing the unchanged blocks is then:

$$\text{REFRESH} = (\text{REPEAT} + 1) \times \text{RATE} - \text{TOTAL}. \tag{3}$$

Static blocks are refreshed in odd/even field pairs. That is, both the odd and the even field portion of a block are refreshed at the same time. The blocks are refreshed so that the "older" blocks are refreshed first. During the first pass, as the change determination is being made, a histogram is computed of the rate required to refresh all the blocks of a certain age. When the first pass is completed and the value of REFRESH is determined, the age threshold THRAGE, is computed so that all blocks older than THRAGE can be sent with less than the REFRESH number of bits. All blocks older than THRAGE are automatically refreshed during the second pass. Blocks with the same age as THRAGE are randomly refreshed until all the remaining rate is depleted. Any blocks that are younger than THRAGE are left unchanged. The ages of all the blocks are then incremented by one for the next frame time.

VIII. SIMULATION OF THE ALGORITHM

The conditional-replenishment system discussed above was simulated on a SEL 32/55 computer system. The test data used consisted of 87 sequential frames of video images taken from commercial broadcast television. Frame 28 from this sequence is shown in Fig. 7. To obtain the data, a video signal was low-pass filtered down to 4 MHz, and then sampled at a rate of 8 Msamples/s. Each sample is at 6 bits of resolution. There are 512 samples per line with 525 lines per frame. Of this, only the visible samples were retained and stored on digital tape, producing images of 416 pels wide and 464 lines high.

To obtain the data, 3 s of video were first recorded on an analog video disc at the normal video rate. The video disc was stepped through the 87 frames at a lower rate so that each frame could be digitized and frozen into a 1-frame digital memory. The frozen frame was transferred to the SEL 32/55 where it was written onto magnetic tapes for future use. To display the results of the simulations, the computer wrote out to a single-frame video display unit and the analog disc recorded one frame at a time.

The simulation was performed by programming the conditional-replenishment algorithm in Fortran on the SEL 32/55. Only the encoder section was simulated, since the image at the decoder is also available at the tracking-frame memory in the encoder. Because the rate buffering was not required for the simulation, it was not programmed. However, to study the performance of the coder in noisy environments, the rate-buffering aspect would have to be simulated. Simulations

Fig. 7. Frame 28 from original video sequence.

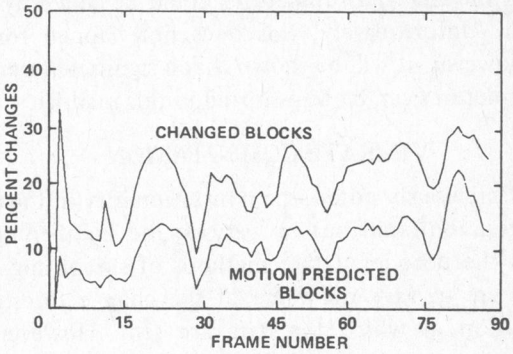

Fig. 8. Variation of changed blocks with time.

were run on the test scene at compressed data rates of 1, 1/2, 1/4, and 1/8 bit/pel.

IX. SIMULATION RESULTS

The transform coding scheme reported in this paper did not produce very much spatial distortion in the coded images. Most of the visible distortion is a result of errors in detecting the changes or, at very low rates, from motion degradation introduced by repeating frames. The majority of errors caused by the change detector were in the "gray wall" areas where there is very little contrast. Where there are edges with large constant changes, change-detector errors do not occur. However, when there are no edges, change-detection errors become quite evident as random fluctuations. These small random changes are missed by the change detector until enough changes have accumulated to trigger a change detection. When this area is replenished, a sudden change in appearance occurs.

In the test scene used, there was considerable variation in the number of changes per frame. This is shown in the curve in Fig. 8 for the 87 frames used. The top curve represents the percent of the changing subpictures as a function of time. The number of changes varied from a low of about 10 percent to a high of 33 percent. A corresponding variation of 3 to 1 would be expected in the data rate required over the 87 frames used.

The lower curve in Fig. 8 depicts the percentage of the total blocks that can be coded in the test scene using motion prediction. Comparing this with the upper curve presents an indication of the effectiveness of the motion-prediction algorithm. These curves demonstrate that 33-50 percent of the changed blocks can be coded using the motion-prediction scheme. The rate can correspondingly be reduced by about 33-50 percent, while still retaining the same fidelity as the higher rate coder without the motion prediction.

When the number of changes becomes too high to send all the data for a frame in one frame time, the decoder repeats the last frame it received until the next frame is completely transmitted. The motion degradation caused by these frame repeats is not too disturbing if just one or two repeats are made. However, if three or more frame repeats are made, the motion becomes very jerky. Table IV below gives the average frame display time for the four rates simulated without motion prediction. At 1 bit/pel there are no frame repeats required, therefore, the average frame display time is 1 frame period. At 1/2 bit/pel about half the frames are repeated once which gives an average display time of 1.56 frame periods. Below this rate, the number of repeats becomes quite high and motion degradation is introduced.

TABLE IV
PERFORMANCE OF THE MOTION-PREDICTION ALGORITHM

AVERAGE RATE (BPP)	AVERAGE FRAME DISPLAY TIME WITHOUT MOTION PRED.	WITH MOTION PRED.
1	1.00	----
1/2	1.56	1.00
1/4	3.30	2.07
1/8	6.62	4.14
1/16	----	9.55

The motion-prediction algorithm was simulated in the test sequence at compression rates of 1/2, 1/4, 1/8 and 1/16 bit/pel. Table IV compares the average frame display times of the conditional-replenishment system with, and without, the motion-prediction algorithm. Enough data could be motion predicted so that, at a rate of 1/2 bit/pel, no frame repeats were required. At 1/4 bit/pel, every frame had to be repeated at least once. This gave an average frame display time of 2.07 frame periods. Below these rates, the frame times became quite high. Comparing this with the frame display times for the other systems shows approximately a 33-percent reduction in frame display time by adding the motion-prediction algorithm.

Fig. 9 compares the average frame display rate of the conditional-replenishment system with, and without, motion prediction. It is apparent the motion-prediction system does perform considerably better. Another interesting fact that can be observed is that the motion-prediction curve is not quite a straight line. As the data rate is reduced, the frame falls slightly faster, especially at the lower rates.

The bottom curve in Fig. 9 shows the performance of the conditional-replenishment system without motion prediction. From this, we can see that there is an almost linear relationship between the bit rate used and the resulting frame rate produced. These frame rates are an average rate over the 89 frames used. The frame rate is extremely dependent on the video images that are being coded, and could either increase

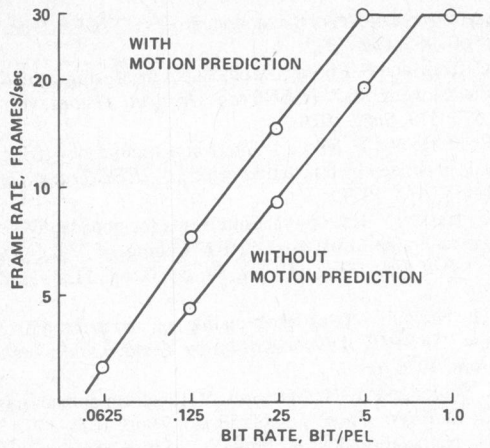

Fig. 9. Coder performance in terms of average frame rate.

Fig. 11. Frame 28 from motion-prediction simulation at 1/2 bit/pel.

Fig. 10. Frame 28 from conditional replensihment at 1 bit/pel.

or decrease depending on the percentage of changes in the images.

The resulting video was viewed at the various compression rates. Fig. 10 shows frame 28 from the condition-replenishment simulation at 1 bit/pel. Because there were no frame repeats done at the 1-bit/pel rate, the video looked quite good. There was no significant difference between this and the original video sequence. At 1/2 bit/pel the frame rate was still high enough to give an acceptable image. This average pel rate corresponds to a data rate of about 3 Mbit/s. At the lower rates, the frame repetition rate becomes quite high and the resulting video is of low quality.

The upper curve of Fig. 9 shows the results of the motion-prediction study. It appears to be quite linear except for the point at 1/16 bit/pel. At this data rate the frame rate is slightly below what would be expected from a linear relationship. This would seem to indicate that the ratio of replenished blocks to motion-predicted blocks is higher at this rate than at the higher frame rates. There are probably two primary causes for this effect. One is that, as the frame rate is reduced, the motion from frame to frame becomes correspondingly higher. It would be expected that the motion predictor would have greater difficulty handling this situation, and a larger percentage of changed blocks would have to be replenished.

Another reason would be that the motion-predicted blocks may contain more coding error, and would therefore increase their changes of requiring replenishment. Generally, the motion-predicted blocks do not appear very different from the transform replenished blocks. However, some blocks do have errors along the block edges because the change detector measures differences evenly throughout the block, and averages them over the entire block. Also, low contrast detail is occasionally lost because of this problem. Fig. 11 shows frame 28 of the 1/2 bit/pel simulation of the motion prediction algorithm. The errors described above can be observed in this image.

X. CONCLUSIONS

Because the motion-prediction system is more efficient and requires a lower data rate, fewer frame repeats are needed. Consequently, a motion-prediction system can run at a lower data rate and still maintain the same frame rate as a compression system witnout motion prediction. The motion-prediction simulation results are quite acceptable down to an average pel rate of 1/4 bit/pel. This corresponds to a transmission rate of 1.5 Mbit/s.

A major problem with the motion-prediction algorithm is that a large number of computations are required to determine the displacement parameters. If the range of allowable displacements were limited, the computational requirements could be drastically reduced. Fig. 12 shows the percentage of motion-predicted blocks versus the size of the horizontal and vertical displacements. If the magnitudes of the displacements were limited to a maximum of three, over 80 percent of the motion predicted blocks could be coded in this way. This would reduce the amount of computations required by a factor of about four from a system allowing the full-range displacements.

It was also determined experimentally that the relationship between the data rate and the frame rate in a conditional-replenishment system is quite linear. By accepting lower frame rates, the transmission rate can be proportionately reduced.

Finally, it may be worthwhile studying the effects of using transforms other than the WHT [20], [21] on the overall performance of the compression scheme presented here.

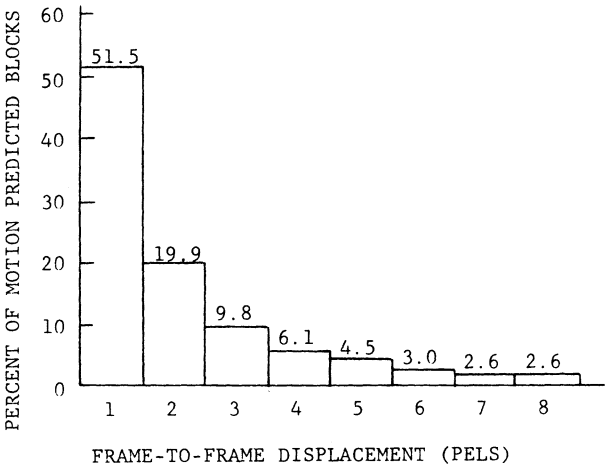

Fig. 12. Distribution of displacement values.

REFERENCES

[1] R. D. Kell, British Patent 341811, 1929.

[2] A. J. Seyler, "Probability distributions of television frame differences," *Proc. IREE Australia,* pp. 355-366, Nov. 1965.

[3] A. J. Seyler and Z. L. Budrikis, "Detail perception after scene changes in television image presentations," *IEEE Trans. Inform. Theory,* vol. IT-11, pp. 31-43, Jan. 1965.

[4] F. W. Mounts, "A video encoding system with conditional picture-element replenishment," *Bell Syst. Tech. J.,* vol. 48, pp. 2545-2554, Sept. 1969.

[5] J. C. Candy, M. A. Franke, B. G. Haskell, and F. W. Mounts, "Transmitting television as clusters of frame-to-frame differences," *Bell Syst. Tech. J.,* vol. 50, no. 6, pp. 1889-1917, July-Aug. 1971.

[6] D. J. Connor, B. G. Haskell, and F. W. Mounts, "A frame-to-frame picturephone coder for signals containing differential quantizing noise," *Bell Syst. Tech. J.,* vol. 52, no. 1, pp. 35-51, Jan. 1973.

[7] F. Rocca and S. Zanoletti, "Bandwidth reduction via movement compensation on a model of the random process," *IEEE Trans. Commun.,* pp. 960-965, Oct. 1972.

[8] C. Cafforio and F. Rocca, "Methods for measuring small displacements of television images," *IEEE Trans. Inform. Theory,* vol. IT-22, no. 5, pp. 573-579, Sept. 1976.

[9] S. Brofferio and F. Rocca, "Interframe redundancy reduction of video signals generated by translating objects," *IEEE Trans. Commun.,* pp. 448-455, Apr. 1977.

[10] B. G. Haskell, "Entropy measurements for nonadaptive and adaptive, Frame-to-Frame, Linear-predictive Coding of Video-telephone Signals," *Bell Syst. Tech. J.,* vol. 54, no. 6, pp. 1155-1174, July-Aug. 1975.

[11] B. G. Haskell, "Interframe coding of monochrome television—A review," in *SPIE, Advances in Image Transmission Techniques,* vol. 87, Aug. 1976, pp. 212-217.

[12] J. A. Stuller and A. N. Netravali, "Transform domain motion estimation," *Bell Syst. Tech. J.,* vol. 58, no. 7, pp. 1673-1702, Sept. 1979.

[13] A. N. Netravali and J. A. Stuller, "Motion-compensation transform coding," *Bell Syst. Tech J.,* vol. 58, no. 7, pp. 1703-1719, Sept. 1979.

[14] A. N. Netravali and J. D. Robbins, "Motion-compensated television coding: Part I," *Bell Syst. Tech. J.,* vol. 58, no. 3, pp. 631-670, Mar. 1979.

[15] H. W. Jones, "A conditional replenishment hadamard video compression," in *SPIE, Applications of Digital Image Processing,* vol. 119, Aug. 1977, pp. 91-98.

[16] H. W. Jones, "Performance of a fixed rate conditional replenishment transform compression," in *NTC '77 Conf. Rec.,* vol. 1, Dec. 1977, pp. 10:1.1-10:1.5.

[17] N. Ahmed, H. Schreiber, and P. Lopresti, "On notation and definition of terms related to a calss of complete orthogonal functions," *IEEE Trans. Electromagn. Compat.,* vol. 15, pp. 75-80, May 1983.

[18] S. C Knauer, "Real-time video compression algorithm for Hadamard transform processing," *IEEE Trans. Electromagn. Compat.,* vol. 18, pp. 28-36, Feb. 1976.

[19] H. F. Harmuth, *Sequency Theory.* New York: Academic, 1978, pp. 163-178.

[20] K. A. Prabhu and A. N. Netravali, "Pel recursive motion compensated color coding," in *Proc. Int. Commun. Conf. (ICC)* (Philadelphia, PA), June 1982, pp. 2G.8.1-G.8.5.

[21] D. Hein and N. A. Ahmed, "On a real-time Walsh-Hadamard/cosine transform image processor," *IEEE Trans. Electromagn. Compat.,* vol. 20, Aug. 1978, pp. 453-457.