Technical Communications

Low-Bit-Rate Coding of Underwater Video Using Wavelet-Based Compression Algorithms

David F. Hoag, Vinay K. Ingle, and Richard J. Gaudette

***Abstract*—Recent advances in autonomous underwater vehicle (AUV) and underwater communication technology have promoted a surge of research activity within the area of signal and information processing. A new application is proposed herein for capturing and processing underwater video onboard an untethered AUV, then transmitting it to a remote platform using acoustic telemetry. Since video communication requires a considerably larger bandwidth than that provided by an underwater acoustic channel, the data must be massively compressed prior to transmission from the AUV. Past research has shown that the low contrast and low-detailed nature of underwater imagery allows for low-bit-rate coding of the data by wavelet-based image-coding algorithms. In this work, these findings have been extended to the design of a wavelet-based hybrid video encoder which employs entropy-constrained vector quantization (ECVQ) with overlapped block-based motion compensation. The ECVQ codebooks were designed from a statistical source model which describes the distribution of high subband wavelet coefficients in both intraframe and prediction error images. Results indicate that good visual quality can be achieved for very low bit-rate coding of underwater video with our algorithm.**

I. Introduction

Future study and exploration of the ocean environment will rely on the use of autonomous underwater vehicles (AUV's) as an alternative to manned submersible missions. They cost far less than manned vehicles, are safer, and can remain submerged for much longer periods of time. Advances in AUV and underwater communication technology have promoted a surge of research activity within the area of signal and information processing. The problem presented in this work involves the transmission of underwater video from an untethered AUV to a remote platform using underwater acoustic telemetry. In the past, an AUV tethered by a fiber-optic cable and equipped with on-board video hardware could capture, process, and transmit video to a surface ship in real time. However, the exploration range of such an AUV is limited by the length of the tether. For the application addressed here, the untethered AUV will be limited only by the distance for which acoustic signaling can provide a reliable channel and the physical constraints on the AUV. Video communication involves the processing and transmission of enormous amounts of information. Thus, prior to transmission, massive data compression must take place on-board the vehicle in order for the acoustic channel to accommodate the application. For example, transmission of monochrome broadcast TV quality video using underwater telemetry would require a compression rate on the order of 1250:1.

The goal of a still-image-coding algorithm is to exploit spatial redundancy among neighboring picture elements (pixels) thereby reducing the size of the two-dimensional (2-D) data set. A popular approach is transform domain coding in which the original data is projected upon a set of basis functions such that a large number of the transform coefficients contain little energy. By zeroing out the coefficients deemed insignificant and efficiently coding the remaining coefficients, the original data set may become compressed. Video compression can be regarded as an extension to the still image compression problem since digital video is composed of a sequence of images. The main difference between the two problems is that video may be compressed at much higher rates since temporal correlation exists among pixels across the frames of the sequence.

A common video-compression implementation is a hybrid of motion-compensated interframe prediction to reduce temporal redundancy and transform coding to reduce intraframe (spatial) redundancy. A coding scheme which employs this type of algorithm is the ITU recommendation H.261 that was developed for real-time encoding of video at integer multiples of 64 Kb/s [1]. The success of the H.261 standard relies upon the energy compaction of the discrete cosine transform (DCT) which is comparable in performance to the mean-square error (mse) optimal Karhunen–Loeve Transform (KLT) [2]. As with the Joint Photographic Expert's Group (JPEG) standard for still-image coding, H.261 reconstructed imagery suffers from visually annoying "blocking" and "contouring" artifacts when the original data is compressed at high rates.

The discrete wavelet transform (DWT) has gained widespread popularity for application to the areas of digital signal processing and communications. It is appealing to the data-compression community for its energy compaction, the multiresolution representation that it provides, and the fast algorithms which can be utilized for implementation of a wavelet decomposition. In [3] and [4], it was shown that a multistage wavelet decomposition combined with model-based standard vector quantization (VQ) or trellis coded vector quantization (TCVQ) could achieve high rates (40-50:1) for compressing underwater images with superior visual quality to JPEG compressed imagery. One reason for the success of wavelet-based algorithms over JPEG is the global transform nature of the DWT which inherently produces no blockiness distortion in reconstructed data. Wavelet-encoded general imagery may exhibit "washed out" or blurred edge areas when compressed at high rates, so care must be taken in coding the high-frequency (detail) subband coefficients. Unlike general image data, underwater images tend to exhibit a large amount of low contrast and low-detail areas. When the data is decomposed by a wavelet filter bank, a majority of the signal energy is compacted into the lowest subband, rendering an efficient representation for high-rate compression. Thus, the objective and subjective improvement over JPEG is even more strikingly apparent when applying wavelet-based coding algorithms to underwater images.

A wavelet-based hybrid video-encoding algorithm for compressing underwater imagery was introduced in [5]. The coder employed the DWT and VQ within the image-compression components, contrasting the DCT-based blocks used by H.261. The results indicated that the algorithm was able to code low-detail underwater video at low bit rates with a high degree of intelligibility in the decoded data. Low-rate coding of high-detail video containing complex motion posed a serious problem to the coder and the quality of the reconstructed images suffered from an unacceptable amount of blocking distortion. The most current implementation of the video coder uses overlapped motion compensation for prediction to reduce these artifacts in a manner which is similar to that presented in [6] and [7]. In addition, ECVQ replaces VQ for obtaining higher compression rates.

Manuscript received November 20, 1995; revised September 6, 1996.

The authors are with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115 USA.

Publisher Item Identifier S 0364-9059(97)01418-0.

Results presented herein demonstrate that these modifications to our original baseline algorithm have significantly improved the overall performance of the video coder.

This communication is composed of the following sections. In Section II, the DWT is introduced from a multiresolution analysis perspective. The realization of a pyramidal wavelet decomposition through the use of a multistage filter bank follows. Section III begins with a brief discussion on three popular approaches to video coding. The section concludes with a high-level presentation of the wavelet-based hybrid video encoder. Implementation issues, including the latest modifications to the coder, are detailed in Section IV, while results on compressing two underwater test sequences are found in Section V. In the final section, a summary on the highlights of this communication followed by plans for future research is presented.

II. Wavelets and Filter Banks

The DWT has recently emerged as a powerful tool for application to a wide variety of areas such as digital signal processing, communications, robot vision, electromagnetics, and applied mathematics. Popular applications include data compression, multiscale detection, and estimation, pattern recognition, and finite element analysis. The underlying mathematics of wavelet theory are extensive, therefore, interested readers are referred to [8]–[12] for more advanced treatments on the subject.

The discrete Fourier transform (DFT) is a well-understood analysis tool that projects a signal upon sine and cosine basis functions that have excellent frequency resolution but poor time resolution. If a signal is composed of few stationary components such as sine waves, the DFT will yield an efficient representation in the transform domain. However, if the signal contains any abrupt changes in time, the DFT will spread the signal out across the entire frequency spectrum, leading to an inefficient representation [13]. Natural imagery typically contains nonstationary elements such as edge structures (sharp luminance transitions in a group of neighboring pixels), thus the use of sinusoidal-based transforms in an image-compression framework is not very appealing.

The DWT employs finite-support basis functions which are well localized in both time and frequency. Wavelet analysis provides good frequency resolution at low frequencies (wide support bases) and good time resolution at high frequencies (narrow support bases). Signals composed of long-duration low-frequency and short-duration high-frequency information can be transformed into a compact set of coefficients in the wavelet domain. Natural images lend themselves well to wavelet-based compression algorithms since such data is characterized by smooth background areas with occasional edges (high spatial frequencies).

A 2-D pyramidal wavelet decomposition is realized by constructing a separable dual channel wavelet filter bank. One stage of a 2-D filter bank is illustrated in Fig. 1.

A 2-D signal $x(m, n)$ is input to the stage and each column of data is low-pass and high-pass filtered by the $h(m)$ and $g(m)$ blocks, respectively, then downsampled by a rate of two. The two resultant 2-D blocks are then fed into the horizontal filtering substage where each row is passed through the LPF and HPF blocks followed by subsampling. This filter bank operation generates four separate subbands that have one quarter of the support of the original signal. The subband LL_1 contains coefficients which are a coarse approximation to the input. Due to the separable structure of the 2-D bank, the detail subbands LH_1, HL_1, and HH_1 contain information which represents horizontally, vertically, and diagonally oriented high-frequency edge structures, respectively. An Nth order 2-D pyramidal decomposition is constructed by connecting the LL_j output from each stage to the input of the subsequent stage, for

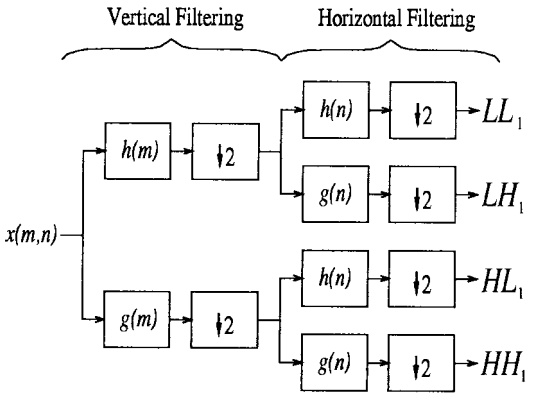


Fig. 1. One stage of a 2-D wavelet analysis filter bank.

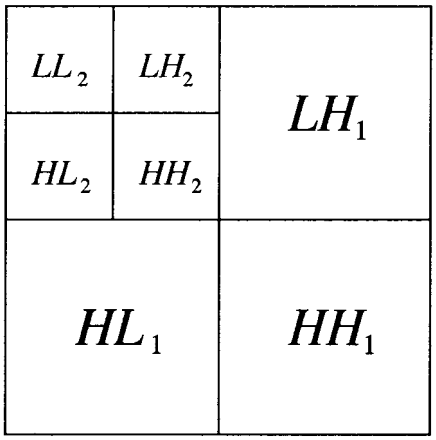


Fig. 2. Frequency partitioning of a 2-stage 2-D pyramidal decomposition.

$1 \leq j \leq N-1$. The spatial frequency partitioning yielded by the 2-stage pyramidal decomposition of a 2-D signal follows in Fig. 2. Due to the excellent energy compaction of the DWT, a majority of the signal energy would reside in the LL_2 subband, especially for low-contrast and detail underwater imagery. Significantly less energy would be contained in the higher subbands, yet important edge structures may be found therein.

Linear-phase FIR filters (symmetric wavelets) are desirable for use in image compression since the amount of induced edge distortion in the presence of coarse quantization is less severe. Additionally, fewer high-frequency artifacts will be induced at the signal boundaries since the signal is mirrored rather than periodized prior to filtering. Linear-phase FIR wavelet filters are designed by relaxing an orthogonality constraint and adopting the use of biorthogonal wavelets, as described in [11]. The relationship between the set of low-pass and high-pass analysis $\{h(n), g(n)\}$ and synthesis $\{\tilde{h}(n), \tilde{g}(n)\}$ filter pairs then becomes

$$\sum_n h(n)\tilde{h}(n+2k) = \delta(k) \tag{1}$$

$$\tilde{g}(n) = (-1)^n h(1-n) \tag{2}$$

$$g(n) = (-1)^n \tilde{h}(1-n) \tag{3}$$

where $\delta(k)$ is the Dirac delta function.

Empirical evidence suggests that the distributions of wavelet coefficients located in the individual subbands of an Mth order decomposition (excluding the LL_M band) of a natural image are well approximated by a generalized Gaussian probability density function

(pdf), as presented in [14]. The pdf is defined as

$$p(x) = \frac{bc}{2\Gamma\left(\frac{1}{c}\right)} e^{-|bx|^c} \quad \text{and} \quad b = \frac{1}{\sigma}\frac{\Gamma\left(\frac{3}{c}\right)^{\frac{1}{2}}}{\Gamma\left(\frac{1}{c}\right)^{\frac{1}{2}}} \tag{4}$$

where σ is the standard deviation of the wavelet coefficients in a given subband and $\Gamma(x)$ is the gamma function. The exponential parameter c dictates the sharpness of the peak located at 0 (e.g., $c = 1 \Rightarrow$ Laplacian, $c = 2 \Rightarrow$ Gaussian). Known source statistics are critical to the design of optimal quantizers. For example, Lloyd–Max scalar quantization depends on a given source model to derive an mse optimal encoder for a defined bit rate. In the design of the quantizers used for this work, the distributions of the high-subband coefficients were assumed to fit a generalized Gaussian model.

III. Video Coding

The goal of any lossy data-compression algorithm is to reduce the amount of information in a given signal while retaining key features in the data set. For imagery, these features include edge structures and background information. Still-image compression is achieved by reducing the spatial redundancy associated with neighboring pixel intensities. The most popular and successful algorithms are transform-based as indicated by the recently adopted JPEG image-compression standard and also the wealth of ongoing image-coding research which employs the DWT. Video data is comprised of a sequence of still images, which when displayed in succession at a sufficiently high frame rate, will recreate a full-motion movie. The sampling frequency of 30 frames/s is typically used to digitize analog video. At such a high rate, there will tend to be a significant amount of temporal correlation among pixels with the same spatial location from frame to frame. Video-compression algorithms should therefore be geared toward reducing both spatial (intraframe) and temporal (interframe) redundancy.

There are three main approaches to coding video data, namely compression of still images, three-dimensional (3-D) transform coding, and motion-compensated hybrid encoding. The first approach is useful in applications such as video editing where any individual frame may need to be accessed for processing without any noticeable waiting time for the user. Low-bit-rate coding is difficult to achieve with this technique since interframe correlation is not considered. Video compression using 3-D transform coding is an area of study in which much research has been conducted, including [15]–[19]. To realize a separable 3-D transform, the rows and columns of a set of still images are first independently transformed, then the resultant data is transformed temporally. Limitations of this approach are the amount of memory and processing speed needed for implementation. For example, a set of 30 monochrome images with a resolution of 512×512 pixels requires a storage capacity of 8 megabytes. A real-time application would be very difficult to implement with the processing speed required to encode such large blocks of data, especially on-board an AUV where power consumption is a critical issue. The last approach, motion-compensated hybrid video coding, combines motion estimation and compensation with transform coding in a DPCM-like fashion, as presented in [5]–[7], [20]. Hybrid video encoding is the most practical approach for the underwater video application since high compression is achievable without a large memory and high-speed processing requirement.

Motion estimation and compensation techniques are either pixel-based or block-based. Pixel-based approaches compute a motion vector for each individual pixel from one frame to the next. These approaches may be highly accurate in estimating complex motion such as translation, rotation, and dilation, however they are computationally intensive. As a result, they are not suitable for real-time encoding of video with current hardware technologies.

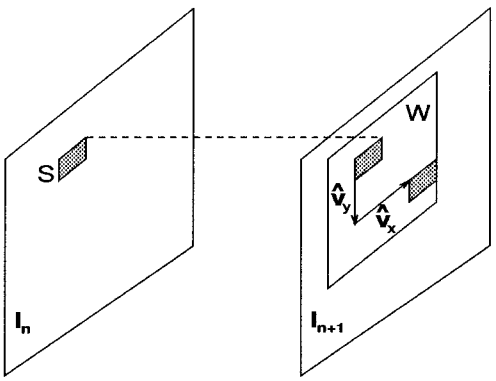


Fig. 3. Block-based motion estimation.

Block-based methods assume a uniform motion for all the pixels within fixed size subblocks. Let $I(x, y, n)$ denote a luminance value at pixel location (x, y) for a discrete time instance n. According to the motion model, a motion vector (v_x, v_y) exists such that

$$I(x + v_x, y + v_y, n + 1) = I(x, y, n). \tag{5}$$

An estimate of the motion vector $(\hat{v}_x, \hat{v}_y)$, which maps the luminance values from a given subblock S within frame I_n to values from a window W within frame I_{n+1}, is found by minimizing the equation

$$(\hat{v}_x, \hat{v}_y) = \arg \min_{(v_x, v_y) \in W} \sum_{(x,y) \in S} \rho(I(x, y, n), I(x + v_x, y + v_y, n + 1)) \tag{6}$$

where $\rho(\alpha, \beta)$ is a distortion function such as mean absolute distance (MAD) or mse. The graphical illustration of this technique is shown in Fig. 3.

One problem associated with block motion estimation is the limitation to uniform translational motion of pixels within a subblock. Also, determination of the window size requires consideration of the tradeoff between motion vector accuracy and computational burden. For example, large windows may be needed for tracking fast object motion at the expense of reducing algorithm speed. Despite the problems linked to these methods, they are appealing for their amenability to hardware implementation and are often used in hybrid video-encoding applications.

The prediction error signals generated when using block-based motion estimation tend to be noisy with structured high spatial frequency information generally located at block boundaries. The 8×8 subblock processing by JPEG in the feedforward path of H.261 distorts the high-frequency residual data, especially in a high-compression-rate scenario. Additionally, the distortion is compounded within the feedback loop and visually annoying artifacts are induced in subsequent images of reconstructed video. An alternative approach to the JPEG compression of the residuals is the use of wavelet-based encoding.

A system diagram of the wavelet-based motion-compensated hybrid video coder is presented in Fig. 4. The coder is implemented with two modes of operation, namely, intraframe coding and prediction-error coding where the former mode refers to compressing a raw image while the latter refers to compressing a prediction-error frame. Note that the encoder is operating during the $(n + 1)$th time slot as indicated by signal labels in the diagram and this represents the error-frame coding mode.

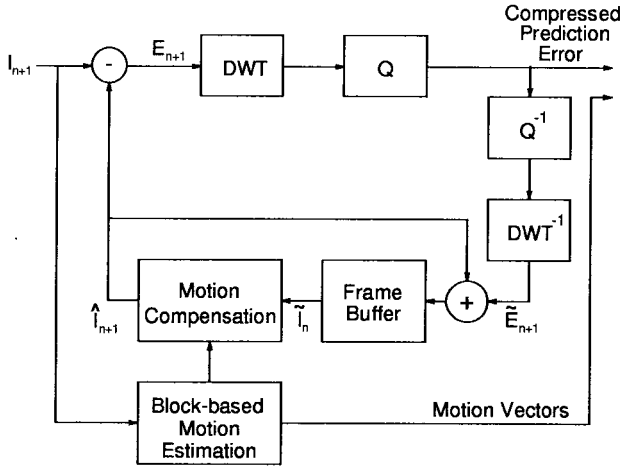


Fig. 4. Wavelet-based video encoder.

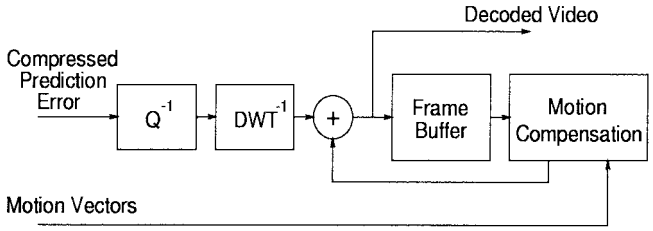


Fig. 5. Wavelet-based video decoder.

To illustrate the functionality of the encoder, a description based on compressing a video sequence of length L follows. Prior to the error-frame coding mode of operation shown in Fig. 4, the first image I_n is intraframe coded using the DWT followed by quantization (Q) in the feed-forward path, and it acts as the reference frame for encoding the remaining $L-1$ frames from the sequence. I_n must be compressed at a relatively low rate (10–20:1), thereby insuring good reconstruction quality not only in this initial frame, but also in the remainder of the sequence. If I_n is compressed at a high rate that yields poor reconstruction quality, then the subsequent images will be decoded with increasingly worse quality. Once I_n is intraframe coded, it is output to the channel and also simultaneously reconstructed by the Q^{-1} and DWT^{-1} blocks of the feedback loop, then stored in the frame buffer as $\tilde{I}_n$. Note that I_n is also stored for one time slot within the motion estimation block. In the next time slot, I_{n+1} is input and block-based motion estimation is used to generate motion vectors associated with each fixed size subblock of I_n. The motion vectors are then combined with $\tilde{I}_n$ using motion compensation to produce a prediction image $\hat{I}_{n+1}$. A prediction error or residual frame is generated as

$$E_{n+1} = I_{n+1} - \hat{I}_{n+1}. \tag{7}$$

The prediction error E_{n+1} is then compressed at a very high rate by the DWT and quantization since it contains little energy. The compressed error frame is transmitted across the channel with its associated motion vector information while the reconstructed residual $\tilde{E}_{n+1}$ is fed back, added to $\hat{I}_{n+1}$ with the result being stored in the frame buffer. The algorithm continues in this manner until L frames are compressed. At the $(n+L)$th frame, the algorithm is reset to encode the next L frames of video data. The video decoder is shown in Fig. 5. Note that it has the same basic structure as the feedback portion of the encoder.

IV. Implementation of the Hybrid Video Encoder

The hybrid video encoder is composed of four major blocks including the DWT, quantization, motion estimation, and motion compensation. Past results on compressing still underwater images and video found in [3]–[5] have demonstrated that coders which use biorthogonal wavelets yield better objective and subjective performance than those employing Daubechies' wavelets. For this reason, the well-known 9-tap and 7-tap biorthogonal wavelet filter pair $h(n)$ and $\tilde{h}(n)$ are employed in the design of the 2-D filter bank. The actual filter coefficients are listed in [14].

Compression of the intraframe and error-frame data was accomplished through the use of vector quantization techniques. According to Shannon's rate-distortion theory, VQ will provide results which are lower-bounded by those attained when applying scalar quantization [21]. Rather than coding individual data elements, VQ codes blocks of data, thus inherently accounting for correlation between neighboring elements. An additional advantage of VQ is that fractional bit rates are provided by the quantizer. Scalar quantizers are constrained to integer rate codebooks, thus offering a more coarse resolution in terms of available bit rate. A VQ codebook is implemented by designing an array of N codevectors with dimension k, corresponding to an average bit rate of $R_{\text{ave}} = \frac{\log_2 N}{k}$ b/sample. In a coding application, both the transmitter and receiver know the values of the codevectors, therefore, only the codebook indexes need to be transmitted.

There are two main issues to consider when implementing a VQ, namely, design of the codebooks and the search method used to find the best "match" to a given source vector. For this work, one codebook design is based upon the well-known LBG algorithm, described in [22]. Each source vector X is quantized using full-search VQ. This approach selects a channel codeword (index) l to represent the vector according to the minimization of

$$l = \arg\min_i \rho(X, C_i) \tag{8}$$

where $\rho(\alpha, \beta)$ is the distortion function (mse for this implementation) and C_i is the ith codevector in the codebook being searched.

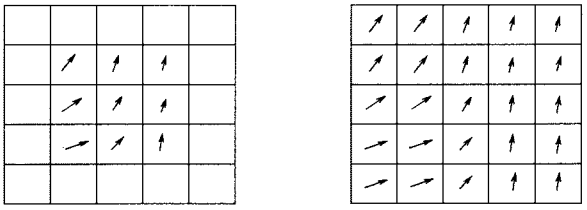
A zero-mean unit variance generalized Gaussian model with exponential parameter $c = 0.7$ was assumed for approximating the distribution of coefficients in the high subbands of the wavelet decomposition. The model was used to generate a set of training data which reflects the statistical properties of the source. A set of 100 000 sample vectors were input to the LBG algorithm, generating codebooks with rates ranging from 0.0625 to 6.0 b/sample.

A wavelet decomposed image is quantized in the following manner. Rates are initially assigned to the subbands using the rate-allocation scheme presented in [3]. Each high subband is individually coded by normalizing the data (subtracting the mean and dividing by the standard deviation) then assigning to it the VQ codebook which has a rate closest to the allocated subband rate.

An improvement to the standard VQ algorithm is ECVQ which was first described in [23]. A source vector X is quantized by selecting a channel codeword which minimizes the equation below:

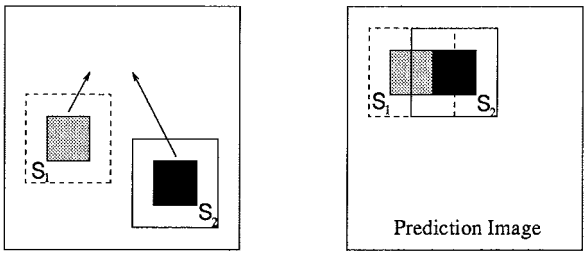
$$l = \arg\min_i \rho(X, C_i) + \lambda \|i\| \tag{9}$$

where $\|i\|$ is the length of the channel codeword representing C_i and λ is a Lagrangian multiplier that remains constant for a given codebook. The effect of this modified cost function can be viewed in the following manner. Given the partition of a sample space of training data, standard VQ assigns to each cell, a constant length channel codeword that represents the cell centroid or codevector. Comparatively, the cell centroids employed in ECVQ are represented by variable-length channel codewords, where the shortest codeword is assigned to the cell which has the largest support (highest probability



(a) (b)

Fig. 6. Zeroth-order prediction of the border subblock motion vectors.



(a) (b)

Fig. 7. Overlapped block-based motion compensation.

of coding a given source vector). Additionally, the nearest neighbor cell is not always the cell selected for coding since a nearby cell with a shorter length channel codeword may be the minimizing solution to the above equation. Graphically, this is equivalent to expanding the size of cells with short codewords and shrinking the size of cells with longer codewords. Details on the general design of the algorithm can be found in [23]. The same pdf model introduced above was used to establish the initial codebooks for training the ECVQ's. A set of 100 000 training vectors were input to the algorithm, generating codebooks ranging from 0.15 to 6.0 b/sample.

The motion estimation algorithm presented for the hybrid video coder in [5] uses block matching with fixed size subblocks of 16×16 pixels and a search area of 30×30 pixels for minimizing the MAD error criterion. A major problem encountered when applying this technique is the generation of inaccurate motion vectors associated with subblocks located at image borders. If a high degree of motion (e.g., vehicle motion) is present within the video sequence, then it becomes very difficult to estimate border subblock vectors without any *a priori* knowledge of the information which lies outside the support of the image. False motion vectors are then computed and a sporadic motion field is generated for the border regions. In turn, the nonsmooth motion field induces blocky structures at the borders of the prediction image resulting in a degradation of the coding performance. The significance of this problem diminishes when making a prediction of the boundary subblock vectors based upon the motion vectors associated with neighboring nonboundary subblocks. A simple zeroth-order predictor was found to work sufficiently well at producing a smooth motion field, as indicated by a large reduction of boundary distortion in the decoded video. The predictor was simply implemented by first computing the inner subblock motion vectors then copying these values to neighboring border subblocks. This operation is illustrated in Fig. 6. Note that the border subblock motion vectors do not have to be transmitted as overhead with those of the inner subblocks.

Nonoverlapped block-based motion compensation may generate high-spatial-frequency regions (blocking edges) located at neighboring subblock boundaries in prediction error images. These occur when subblocks combined with their respective motion vectors point to the same location or when no motion vector points to a given location in a prediction image. At high compression rates, wavelet-based compression algorithms will smooth out the high-frequency information, inducing some distortion in the decoded video.

Overlapped block-based motion compensation was presented in [6] and [7] as a means of reducing the amount of blocky regions that occur in the prediction-error images. By applying this technique, the motion vectors associated with each subblock from a partitioned image are computed in the conventional manner previously described. However, the subblocks used for motion compensation are enlarged and will overlap as illustrated in the prediction image of Fig. 7. Note that the shaded blocks represent the subblocks that are used for motion estimation. When the enlarged subblocks are projected upon potentially new locations, the likelihood of pixels occupying the same location increases. The luminance values of pixels that are in contention for the same spot are averaged together. This operation results in a local smoothing of the edge structures in the prediction image and residual as well. Additionally, the use of enlarged subblocks in motion compensation reduces the incidence of uncovered areas in the prediction. The combination of local edge smoothing and the reduction of uncovered areas in the residual enables wavelet-based algorithms to achieve much higher coding efficiency.

V. Video Compression Results

For the compression results presented here, two different sequences of monochrome underwater video data (frame size 256×256 pixels) were selected. Test sequence 1 (Titanic) contains 10 frames that exhibit low contrast and detail. The only motion present in the video is that due to the underwater vehicle. The video was sampled at 30 frames/s and the vehicle is moving at a slow rate, thus the motion is relatively easy to track using block-based motion estimation. Test sequence 2 (Cortez) is composed of 50 frames that contain much more detail such as rock formations and tubeworms. Two types of motion are present in the video including vehicle motion and the independent motion of a tubeworm. The video was sampled at a rate of 10 frames/s. Due to the high detail, complex motion and lower sampling rate, test sequence 2 is much more difficult to compress.

The peak signal-noise ratio (PSNR) is an objective measure which is commonly used to assess the performance of image and video compression algorithms. It is defined as

$$\mathrm{PSNR} = 10\log_{10}\frac{\alpha}{\sigma_\epsilon^2} \tag{10}$$

where α is the squared peak-to-peak value of the pixels in the original image ($\alpha = 255^2$ for 8-b/pixel monochrome imagery) while σ_ϵ^2 is the sample error variance between the original and reconstructed image. Typically, PSNR values from 25 db up to 35 db represent a scale of subjective visual quality ranging from poor to excellent. It should be noted that the PSNR does not always reflect the subjective quality of reconstructed images since the measure does not account for any characteristics of the human visual system (HVS).

In generating all results, the initial frame was compressed at approximately 20:1 using the biorthogonal DWT with four levels of decomposition followed by quantization. The subsequent residual images contain significantly less energy than a raw data image, therefore, they can be compressed at very high rates, on the order of 400–500:1. For this implementation, seven levels of wavelet decomposition were used by the wavelet coder to achieve these high compression rates. The motion vectors associated with the subblocks of each frame contribute to the overall bit rate as well. The motion estimation algorithm utilizes subblocks with size 16×16 pixels and the number of bits/motion vector is 4 (up to ±7 pixel translations in the x, y directions) corresponding to an overhead of 196 bytes

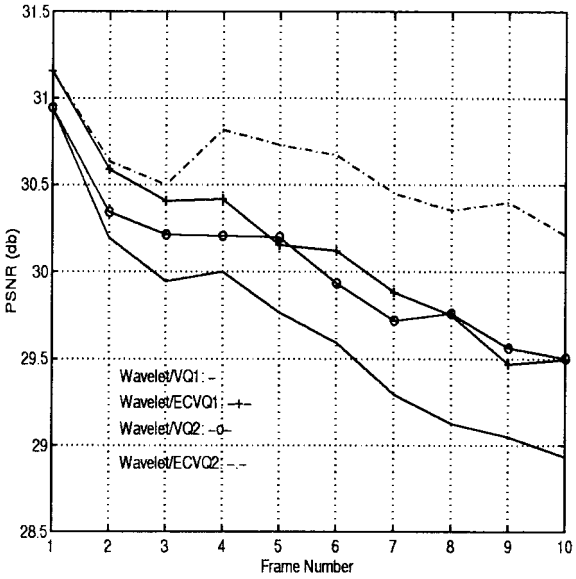


Fig. 8. PSNR versus frame number for the Titanic sequence.

TABLE I
RESULTS SET NAMES

Results Name	Zero Order Estimation of Border Subblock Motion Vectors	Overlapped Motion Compensation
Wavelet/VQ1	no	no
Wavelet/ECVQ1	no	no
Wavelet/VQ2	yes	yes
wavelet/ECVQ2	yes	yes

per residual frame since border subblock motion vectors are not transmitted. Other cases were tested for decreasing the block size and increasing the window size (subblock search area) in the motion estimator but the gains were insignificant at the expense of increasing the computational load. The non-overlapped motion-compensation algorithm employs subblocks of 16×16 pixels while those used in the overlapped motion compensation are of size 32×32 pixels.

For each test sequence, four sets of results were generated. The set name descriptions are located in Table I.

The plots of PSNR versus frame number for the compression of the Titanic sequence are shown in Fig. 8. Note the decreasing slope of the distortion curve as the frame number increases. This trend warrants the need for periodically resetting the compression algorithm, beginning with high-quality intraframe coding.

The algorithm which used ECVQ coded the Titanic video at an average rate of 0.08 b/pixel (100:1 compression) while that which used VQ coded the data at a higher rate of 0.09 b/pixel (90:1 compression). It is clear that quantization using ECVQ gives an improvement in PSNR over VQ. An additional gain in PSNR is also achieved when using the zeroth-order estimation of border subblock motion vectors combined with overlapped motion estimation. The overall improvement in PSNR from the results of wavelet/ECVQ2 versus wavelet/VQ1 is significant. If the coding rates of the two algorithms were equal then the performance of wavelet/ECVQ2 would be even better yet.

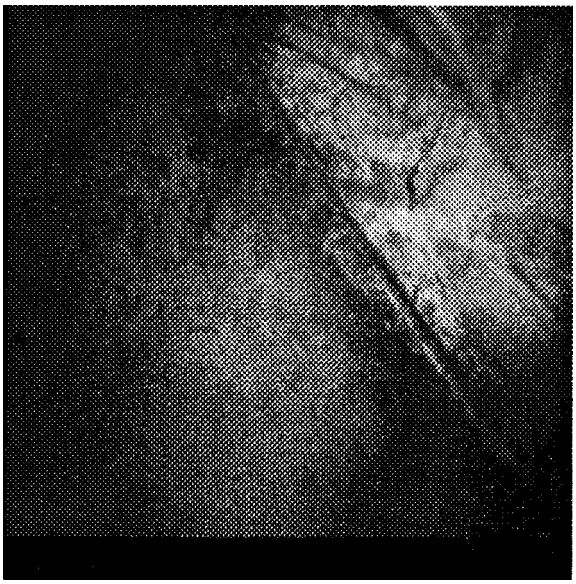


Fig. 9. Original image #10 from Titanic video.

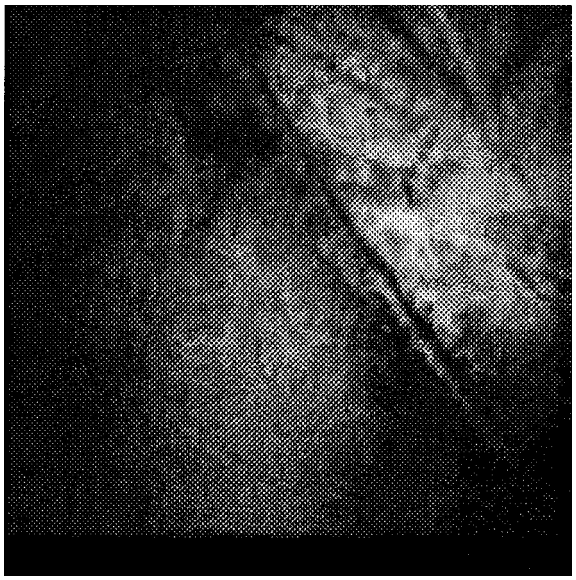


Fig. 10. Wavelet/VQ1 reconstructed image.

In Fig. 9, the tenth image from the original Titanic sequence is displayed. This image was chosen for subjective comparison to the results since it is the last image in the sequence and therefore corresponds to the most degraded image. Figs. 10 and 11 show the reconstructed Titanic images for the wavelet/VQ1 and wavelet/ECVQ2 results, respectively. Both images indicate that some of the edge structures have been slightly blurred yet the overall quality is good for such high rate compression. One difference between the two results is that the wavelet/VQ1 image exhibits some blocky artifacts due to the non-overlapped motion compensation. When the wavelet/VQ1 images are played back as a movie, the motion appears to be choppy especially in the background areas where pixel intensities are more uniform. The overlapped motion compensation used for the wavelet/ECVQ2 case effectively removes the artifacts, resulting in a much more visually pleasing motion flow.

The PSNR versus frame number plots for the results on compressing the Cortez sequence follow in Fig. 12. The algorithms are reset at every tenth frame, indicated by large PSNR jumps at these points

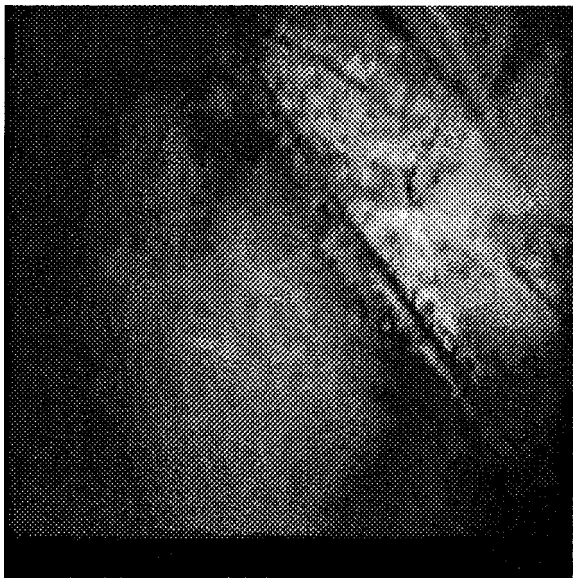


Fig. 11. Wavelet/ECVQ2 reconstructed image.

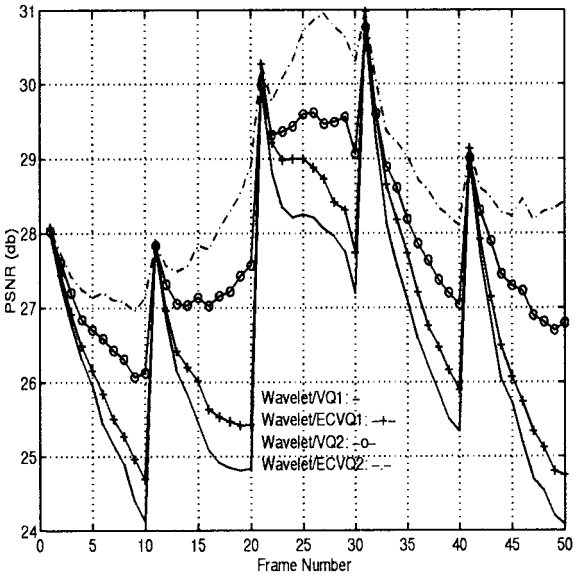


Fig. 12. PSNR versus frame number for the Cortez sequence.

on the curve. With exception to the results of wavelet/ECVQ2, all the curves drop off at a relatively fast rate after an image has been intraframe-coded. This is not a surprising trend due to the high detail and complex motion in the sequence.

All the sequences were coded with an average rate of 0.145 b/pixel (55:1 compression). These results also support the Titanic sequence findings regarding the improvements which can be attained in terms of PSNR. However, for this case, the gain in performance is much greater for wavelet/ECVQ2 as opposed to wavelet/VQ1. Fig. 13 shows the 50th image from the original Cortez sequence while the reconstructed images of wavelet/VQ1 and wavelet/ECVQ2 are found in Figs. 14 and 15.

The reconstructed data of wavelet/VQ1 is highly distorted due to blocking artifacts located throughout the image. This is attributed to a high degree of global motion in the video that initially flows quickly to the right then reverses direction and flows to the left. It is extremely difficult to predict the data coming into view from outside the support of the images. As a result, the error images contain large amounts of energy and when compressed at high rates, significant

Fig. 13. Original image #10 from Cortez sequence.

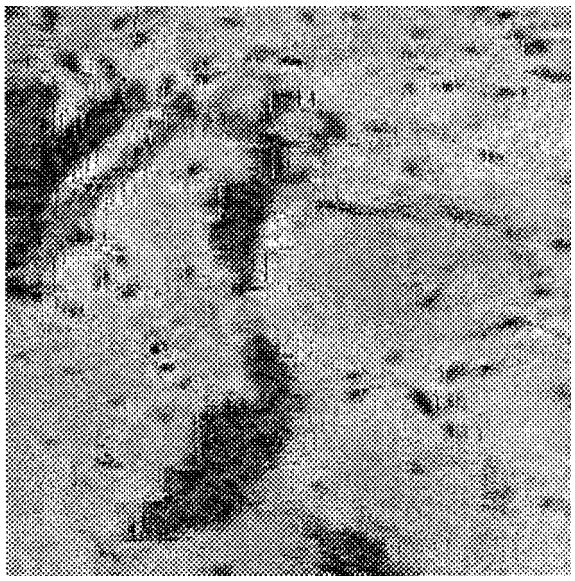


Fig. 14. Wavelet/VQ1 reconstructed image.

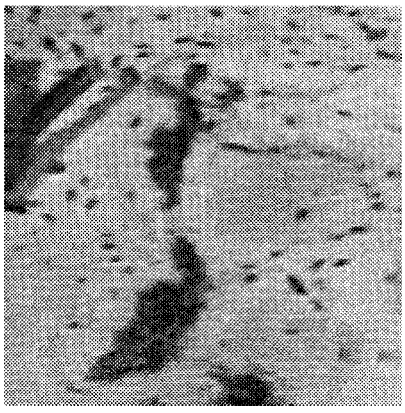


Fig. 15. Wavelet/ECVQ2 reconstructed image.

distortion is induced in the decoded imagery. The error accumulates in the feedback loop of the encoder, resulting in increasingly poor performance. The quality of the wavelet/ECVQ2 image is far superior

to the wavelet/VQ1 result. The zeroth-order prediction of the border subblock motion vectors from the inner subblock vectors is successful at generating a smooth motion field. The combination of this with the overlapped motion compensation produces a prediction that is free of block structures not only within the image but also at the border regions. In turn, the coding efficiency increases since higher quality images can be compressed at rates comparable to those of the wavelet/VQ1 case.

VI. Summary of Work and Future Research

In this communication, we initially presented a wavelet-based hybrid video-compression algorithm that uses standard VQ for coding underwater imagery at very low bit rates. The coder was successful at providing good reconstruction quality for low-rate coding of underwater video which exhibits a low degree of detail and simple motion. However, significant blocky distortion was induced in the reconstructed video when coding high-detail imagery with complex motion at low rates. The problem was linked to the inaccurate estimation of motion vectors associated with border subblocks in the images. This problem was addressed by employing a zeroth-order prediction of the border subblock motion vectors based upon previously computed inner subblock motion vectors. The combination of this approach with overlapped motion compensation produced smoother prediction images, thereby reducing the residual energy and improving the overall coding efficiency. The coding performance of the algorithm was further enhanced by the replacement of standard VQ with ECVQ. The video coder has been applied to compressing general image sequences as well as underwater imagery and experimental results have consistently indicated that higher performance is attainable when coding underwater image sequences. In conclusion, these findings support our claim that the coder works particularly well at exploiting the low contrast and detailed nature of underwater image data.

Due to the simple rate allocation strategy used by our compression algorithm, it is difficult for the actual encoding rate to meet a preset target rate. This same problem also prevents our coder from attaining higher compression rates than those that were cited in the results of this communication. For future work, we will investigate the use of more "appropriate" rate allocation schemes. The problem will involve the minimization of an objective cost function under the constraint of a given bit budget. Our goal is to achieve higher compression with comparable quality to the video which our current coder can provide at its highest rate.

Another research direction which should boost our compression gains is entropy coding of motion vector information. There is generally a high degree of correlation among motion vectors associated with neighboring subblocks, especially in the presence of global motion activity such as that which may be attributed to the motion of an AUV.

References

[1] CCITT Recommendation H.261, "Video codec for audio visual services at $p \times 64$ Kbits/s," COM XV-R 37-E, 1990.

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

[3] D. Hoag and V. Ingle, "Underwater image compression using the wavelet transform," in *IEEE Proc. Oceans 94*, Brest, France, Sept. 1994, pp. 533–537.

[4] D. Hoag, H. Arda Aksu, V. K. Ingle, M. Salehi, and J. G. Proakis, "Underwater image compression using the wavelet transform with trellis coded vector quantization," in *Proc. ICSPAT 95*, Boston, MA, Oct. 1995.

[5] D. Hoag and V. K. Ingle, "Underwater video compression using the wavelet transform," in *Proc. Oceans 95*, San Diego, CA, Sept. 1995.

[6] M. Ohta and S. Nogaki, "Hybrid picture coding with wavelet transform and overlapped motion-compensated prediction coding," *IEEE Trans. Signal Processing*, vol. 41, pp. 3416–3424, Dec. 1993.

[7] D. G. Sampson, E. A. B. da Silva, and M. Ghanbari, "Low bit-rate video coding using wavelet vector quantisation," in *Proc. Inst. Elect. Eng. Vis. Image Signal Processing*, vol. 142, no. 3, pp. 141–149, June 1995.

[8] G. Strang, "Wavelets and dilation equations: A brief introduction," *SIAM Rev.*, vol. 31, no. 4, pp. 614–627, Dec. 1989.

[9] ——, "Wavelet transforms vs. fourier transforms," *Bull. Amer. Math. Soc.*, vol. 28, no. 2, pp. 1–14, Apr. 1993.

[10] C. K. Chui, *An Introduction to Wavelets.* New York: Academic, 1992.

[11] I. Daubechies, *Ten Lectures on Wavelets*. Philadelphia, PA: SIAM, 1992, no. 61.

[12] M. Vetterli and J. Kovacevic, *Wavelets and Subband Coding.* Englewood Cliffs, NJ: Prentice-Hall, 1995.

[13] O. Rioul and M. Vetterli, "Wavelets and signal processing," *IEEE Signal Processing Mag.*, vol. 8, pp. 14–38, Oct. 1991.

[14] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transforms," *IEEE Trans. Image Processing*, vol. 1, pp. 205–220, Apr. 1992.

[15] G. Karlsson and M. Vetterli, "Three dimensional subband coding of video," in *Proc. ICASSP*, Albuquerque, NM, Apr. 1990.

[16] K. M. Uz, M. Vetterli, and D. J. LeGall, "Interpolative multiresolution coding of advanced television with compatible subchannels," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 1, pp. 86–99, Mar. 1991.

[17] K. N. Ngan and W. L. Chooi, "Very low bit rate video coding using 3-D subband approach," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 309–316, 1994.

[18] D. Taubman and A. Zakhor, "Multirate 3-D subband coding of video," *IEEE Trans. Image Processing*, vol. 3, pp. 572–588, 1994.

[19] C. I. Podilchuk, N. S. Jayant, and N. Farvardin, "Three-dimensional subband coding of video," *IEEE Trans. Image Processing*, vol. 4, pp. 125–139, Feb. 1995.

[20] M. I. Sezan and L. J. Lagendijk, *Motion Analysis and Image Sequence Processing.* Boston, MA: Kluwer Academic, 1993.

[21] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression.* Boston, MA: Kluwer Academic, 1992.

[22] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. 28, pp. 88–99, 1980.

[23] R. M. Gray, P. Chou, and T. Lookabaugh, "Entropy constrained vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 31–42, 1989.