Journal of Mathematical Imaging and Vision 7, 41–53 (1997)

Image Compression and Transmission Through a Low-Rate Ultrasonic Link in Subsea Telerobotic Applications

SHI YIN
Dept. of Mechanical and Industrial Engineering, University of Toronto, Toronto, Ontario M5S 3G9, Canada

JENS G. BALCHEN
Dept. of Engineering Cybernetics, The Norwegian Institute of Technology, N7034, Trondheim, Norway

Abstract. This paper describes a progressive data compression algorithm for subsea gray-level image (data) transmission through a low rate ultrasonic link in a telerobotics system. The proposed image compression algorithm is based on JPEG, and has been modified for a specific subsea application, where the communication bit rate is confined to 100–200 bits/s and a frequent updating of reconstructed images is required.

The experimental result based on 23 real images shows that the proposed image data compression algorithm performs better than JPEG when images are reconstructed from a small amount of data. The transmission error effect and computational complexity have also been analysed with respect to the proposed algorithm.

Keywords: gray-level image, progressive image compression, ultrasonic link, telerobotic application

1. Introduction

Image compression and transmission are extremely useful in telerobotic applications, because the human operator depends to a great extent on visual information. This paper presents a progressive image compression algorithm, which is specially designed for transmitting image data through a low-rate ultrasonic link in subsea telerobotic application.

The research reported here was conducted in close collaboration with a subsea telerobotics research program at the Norwegian Institute of Technology, named *MOdel-BAsed TELeoperation of a subsea vehicle over a narrow band communication link* (MOBATEL) [1]. The objective of the MOBATEL program is to develop a free swimming (untethered) underwater vehicle with 6 degree-of-freedom, capable of operating at great depths for such applications as inspection, data collection, handling of objects, teleoperated assembly of equipment and simple machining operations, among others. The main problem to be dealt with in MOBATEL is the elimination of any cable for communication, thus confining the communication channel to an ultrasonic link with a rate of approximately 100–200 bits/s.

There are many image compression algorithms available [2], ranging from lossless to lossy; from spatial domain to transform domain. Image compression algorithms can also be classified according to the image to be processed, such as colour or gray-level images, still images or moving image sequences.

When an image sequence is coded, the redundancy between image frames can be further reduced, in addition to the redundancy reduction between pixels. In MOBATEL, due to the narrow communication bandwidth and consequently the low frame rate, successive image frames will differ significantly. Consequently, techniques for compressing moving image sequences are of low utility due to the low interframe correlation.

When compressing and transmitting a still image, a progressive algorithm enables an approximate image to be built up quickly and the details to be transmitted progressively. In MOBATEL, subsea images must be transmitted to a human observer over a 100–200 bits/s ultrasonic link. Obviously, since the transmission time is long and a quick operator interaction is required, a progressive compression technique is useful.

In this paper a new progressive image compression algorithm, MOPIC (MOBATEL Progressive Image

Compression) algorithm is presented, which emphasizes:

(1) *The reconstruction of images from a few data*. This is desired by the narrow bandwidth limitation and fast system response requirement. We especially emphasize the reconstruction of images from several hundred bytes of data.
(2) *Updating the reconstructed image with a small amount of data*. For visual communication, a slow channel generally refers to a typical bandwidth of 64 k bits/s (e.g., a telephone line). When a progressively compressed image is transmitted through such a channel, 2 k bits of data per updating, which is equivalent to an updating frequency of 32 Hz, is sufficient for human interaction. Higher frequency will not improve the operational performance. In MOBATEL, less data per updating is required. For example, a 2 k bits updating is equivalent to a frequency of 0.05 Hz on a 100 bits/s channel, which obviously is too low for a human operator to properly interact with the system.

2. The MOPIC Algorithm

The MOPIC algorithm is illustrated in Fig. 1. In general, a 256×256-pixel image is used as a source image and it is sent to a Forward Discrete Cosine Transform (FDCT). The FDCT coefficients are first quantized by a coarse quantizer. The output of the first quantizer is entropy coded and transmitted.

With a finer quantization table, the second quantizer quantizes the residue from the first one. The output of the second quantizer is also entropy coded. The third quantizer takes the residue from the second one, and so on for the ith quantizer. For transmission, the coded data from the ith quantizer follow the coded data from the $(i-1)$th quantizer in a data stream.

2.1. The FDCT and Quantization

The FDCT is given as following [3]:

$$C(u,v) = \alpha(u)\alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cdot \cos\left[\frac{(2x+1)u\pi}{2N}\right] \cdot \cos\left[\frac{(2y+1)v\pi}{2N}\right] \tag{1}$$

where $f(x,y)$ is a source image, $x, y = 0, 1, 2, \ldots, N-1, (N = 256)$

$$\alpha(t) = \sqrt{\frac{1}{N}} \quad \text{for } t = 0$$

$$\alpha(t) = \sqrt{\frac{2}{N}} \quad \text{for } t = 1, 2, 3, \ldots, N-1$$

Following the FDCT is the quantization process. A quantizer accepts the output of the FDCT, or the residue from the previous quantizer (Fig. 1). Each of the 256×256 input elements is uniformly quantized via a quantization table. A midtread uniform quantizer is used, as shown in Fig. 2(a). Since most high frequency FDCT coefficients are very small and produce zero outputs via a midtread quantizer [4], we thus avoid large amounts of high frequency noise at the early stage of image reconstruction.

In particular, quantization is defined as the division of each input element by its corresponding quantization step size, followed by a rounding to the nearest integer, as shown in the following equation:

$$F^Q(x,y) = \left[\frac{F(x,y)}{Q(x,y)}\right] \tag{2}$$

where x, y are coordinates, $Q(x,y)$ is quantization step size, $F(x,y)$ and $F^Q(x,y)$ are input and output values of a quantizer, and [ ] denotes rounding to the nearest integer.

All the quantizers use different quantization tables, and each quantization table is used in both quantization and de-quantization process (Fig. 1). In principle, the design of quantization tables in MOPIC takes the following rules into account:

(1) The quantization tables are rotation invariant around the DC coefficient. That is the quantization step value at position (x,y) depends only on $\sqrt{x^2+y^2}$. A coarse quantization step at (x,y) will introduce large quantization distortion at the frequency and direction indicated by (x,y). Note that since the subsea vehicle in MOBATEL has 6 degree-of-freedom, images can be acquired at various camera orientations. And also because these images are square, the frequency distribution of image distortions can not be specified a priori. We therefore justify the isotropic distribution of quantization error with respect to image coordinates.

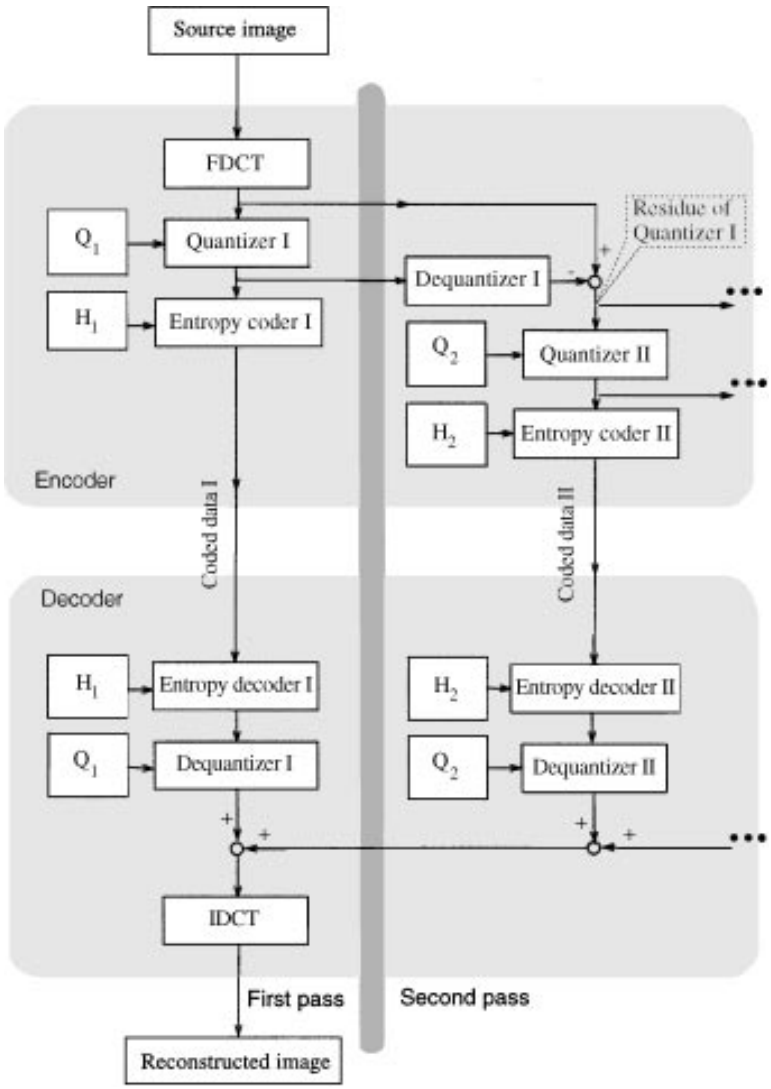


Figure 1. An overview of the MOPIC algorithm.

(2) Low frequency coefficients are quantized more finely than high frequency coefficients, since human eyes are more sensitive to the variation of low frequency [5].
(3) The coarsest quantization step size must be large enough, and the difference in step sizes between two successive quantization processes must be small enough to guarantee that the quantizer outputs will not be so large that they exceed the input range of the following entropy coding process.

In practice, based on extensive experiments with underwater images, a simple quantization table design is given as:

$$Q_1(x, y) = 100 \cdot \left(\sqrt{x^2 + y^2} + 1\right) \quad (3a)$$

$$Q_2(x, y) = 10 \cdot \left(\sqrt{x^2 + y^2} + 1\right) \quad (3b)$$

$$Q_3(x, y) = \sqrt{x^2 + y^2} + 1 \quad (3c)$$

where $Q_1(x, y)$, $Q_2(x, y)$, and $Q_3(x, y)$ are step size values of quantization tables at position (x, y) in the first, second, and third quantization passes respectively. More quantization passes can be assigned if higher image quality is required.

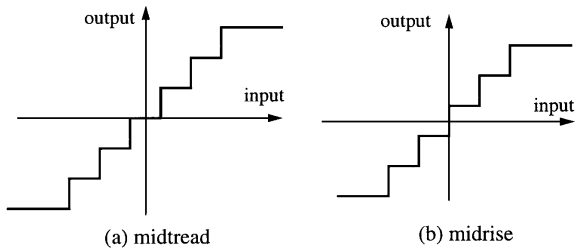


Figure 2. Uniform quantizer characteristic.

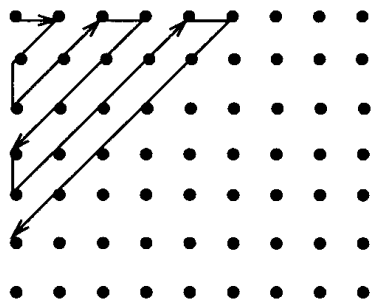


Figure 3. The zig-zag sequence employed in scanning the transformed image. Note that the origin is at the upper left.

2.2. The Zig-Zag Sequence and Entropy Coding

After quantization, the quantized coefficients are scanned according to the "zig-zag" sequence [6], as shown in Fig. 3. This ordering facilitates entropy coding by scanning low-frequency coefficients before high-frequency coefficients. Hence the principal coefficient can be transmitted and recovered first.

The final step in the encoder process is coding, where lossless compression is achieved by encoding the quantized FDCT coefficients more compactly based on their statistical characteristics. In MOPIC, the entropy coding consists of two steps, similar to JPEG [7]. The first step converts the zig-zag sequence of quantized coefficients into an intermediate sequence of symbols. The second step converts the symbols into a compressed data stream.

In the intermediate sequence, each non-zero coefficient is represented in combination with the "runlength" (consecutive number) of zero-valued coefficients which precede it in the zig-zag sequence. Each such runlength/non-zero-coefficient combination is represented by a pair of symbols:

symbol-1	symbol-2
(RUNLENGTH, SIZE)	(AMPLITUDE)

Symbol-1 represents two pieces of information, RUNLENGTH and SIZE. Symbol-2 represents a single piece of information designated AMPLITUDE, which is simply the amplitude of the non-zero coefficient. RUNLENGTH is the number of consecutive zero-valued coefficients in the zig-zag sequence preceding the non-zero coefficient being represented. SIZE is the number of bits used to encode AMPLITUDE.

In MOPIC, each symbol-1 is encoded with a Variable-Length Code (VLC) from a Huffman table, which is predetermined based on the statistics of the quantized FDCT coefficients. The Huffman tables, one for each quantization pass, are generated using the procedure introduced in [5]. Each symbol-2 is encoded with a "Variable-Length Integer" (VLI) code. The structure of the symbol-1 and symbol-2 intermediate representation, as well as the detail VLC and VLI coding procedures, are discussed in [5] and [7].

2.3. Entropy Decoding, Dequantization and Inverse DCT

In MOPIC's image reconstruction process, the entropy decoding and dequantization are the inverse processes of the entropy coding and quantization in the compression stage. Identical Huffman and quantization tables are used in both image compression and reconstruction processes. The entropy decoding and dequantization processes are shown in Fig. 1.

An example of DCT and quantization is illustrated in Fig. 4. Note that Fig. 4 describes only the first quantization pass and the quantization table is generated by Eq. (3a). In the second quantization pass, the quantizer accepts the residue of the quantizer in the first pass, that is the difference of the FDCT coefficients and the denormalized quantized coefficients as shown in Fig. 4.

Following entropy decoding and dequantization is the Inverse DCT (IDCT). A special iterative IDCT scheme is used to achieve a higher refresh frequency

(a) a source image sample

-4284	4410	-1989	-103	-394	137	...
972	994	1311	-1002	755	445	...
-4162	-4327	-1017	-108	1448	26	...
8	-3611	-717	372	402	-317	...
-88	-1523	747	601	-248	-597	...
-95	-1046	1508	1265	-849	117	...
:	:	:	:	:	:	

(b) FDCT coefficients

100	200	300	400	500	600	...
200	241	323	416	512	609	...
300	323	382	460	547	639	...
400	416	460	524	600	683	...
500	512	547	600	665	740	...
600	609	638	683	740	807	...
:	:	:	:	:	:	

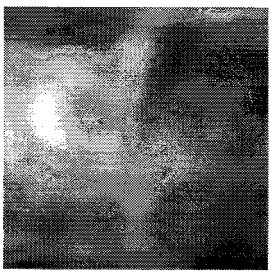
(c) quantization table

-43	22	-7	0	-1	0	...
5	4	4	-2	1	1	...
-14	-13	-3	0	3	0	...
0	-9	-2	1	1	0	...
0	-3	1	1	0	-1	...
0	-2	2	2	-1	0	...
.	.	.	.	.	.	

(d) normalized quantized coefficients

-4300	4400	-2100	0	-500	0	...
1000	964	1292	-832	512	609	...
-4200	-4199	-1146	0	1641	0	...
0	-3744	-920	524	600	0	...
0	-1536	547	600	0	-740	...
0	-1218	1276	1366	-740	0	...
:	:	:	:	:	:	

(e) de-normalized quantized coefficients



(f) the reconstructed image sample

Figure 4. DCT and quantization examples.

for image reconstruction. In particular, the IDCT is given as following [3]:

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\alpha(u)\alpha(v)C(u,v) \cdot \cos\left[\frac{(2x+1)u\pi}{2N}\right]\cdot\cos\left[\frac{(2y+1)v\pi}{2N}\right]. \tag{4}$$

Assume that at time T the recovered DCT coefficients and the reconstructed image are denoted by $C_T(u,v)$ and $f_T(x,y)$ respectively. At time $T' = T + \Delta T$, when one more intermediate symbol pair is received, the recovered DCT coefficients and the reconstructed image are denoted by $C_{T'}(u,v)$ and $f_{T'}(x,y)$ respectively. From the preceding assumption, we have the following equation:

$$C_{T'}(u,v) = C_T(u,v) + \triangle(u,v), \tag{5}$$

where $\Delta(u,v)$ contains only one non-zero element d at position (m,n), which is determined by the previous entropy decoding step.

Therefore, we have:

$$\begin{aligned} f_{T'}(x,y) &= \sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\alpha(u)\alpha(v)C_{T'}(u,v) \cdot\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cdot\cos\left[\frac{(2y+1)v\pi}{2N}\right] \\ &= \sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\alpha(u)\alpha(v)C_T(u,v) \cdot\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cdot\cos\left[\frac{(2y+1)v\pi}{2N}\right] \\ &\quad + \sum_{u=0}^{N-1}\sum_{v=0}^{N-1}\alpha(u)\alpha(v)\Delta(u,v) \cdot\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cdot\cos\left[\frac{(2y+1)v\pi}{2N}\right] \end{aligned} \tag{6}$$

Since $\Delta(u,v)$ contains only one non-zero element d at position (m,n), we can obtain the following IDCT

scheme for progressive image reconstruction:

$$f_{T'}(x,y) = f_T(x,y) + \alpha(m)\cdot\alpha(n)\cdot d \cdot \cos\left[\frac{(2x+1)m\pi}{2N}\right]\cdot\cos\left[\frac{(2y+1)n\pi}{2N}\right]. \quad (7)$$

In iterative structure, we have:

$$f_k(x,y) = 0 \quad k = 0 \quad (8a)$$

$$f_k(x,y) = f_{k-1}(x,y) + \alpha(m_k)\cdot\alpha(n_k)\cdot d_k \cdot \cos\left[\frac{(2x+1)m_k\pi}{2N}\right]\cdot\cos\left[\frac{(2y+1)n_k\pi}{2N}\right] \quad k = 1,2,3,4,\ldots \quad (8b)$$

where k is the number of received intermediate symbol pair, d_k is the value of the non-zero DCT coefficient in the kth intermediate symbol pair, and m_k and n_k are its coordinates respectively.

2.4. Summary

The MOPIC algorithm is progressive and DCT-based. In principle, it follows the Baseline method of JPEG [7] with the following modifications:

(1) In JPEG, an 8×8 quantization table, which can be generated for each image, is included in the compressed data stream. Such a feature can improve image quality in the final stages of progressive transmission since the quantization takes statistical features of the image into account. MOPIC uses a fixed 256×256 quantization table in order to achieve better image quality at the outset of the image reconstruction process.
(2) In order to make the minimum meaningful data unit small and achieve high update frequency consequently, MOPIC simultaneously employs successive quantization and progressive frequency updating. That is, the low frequency coefficients (to which human eyes are more sensitive) are given higher priority for transmission than high frequency coefficients, and the more significant bits are transmitted first.
(3) In MOPIC, a 256×256-pixel image is processed as a single image block. In contrast, JPEG divides an image into 8×8-pixel blocks. This modification reduces the correlation between image blocks. Additionally, by processing the entire image as one block, one avoids JPEG's blocking effect, which degrades the quality of images reconstructed from a few data, as shown in Fig. 6.

3. Experiment

The MOPIC algorithm, which is presented in the previous section, was tested on 23 real images, including both underwater and land images. The test images are shown in Fig. 5.

The experimental quantization tables are given by Eq. (3), while the Huffman tables are obtained using the statistics of the test images. Experimental results are recorded in Table 1 in terms of data size for image

Table 1. The SNR values obtained from the MOPIC algorithm.

Data/ image	100 bytes	200 bytes	400 bytes	800 bytes	1600 bytes	3200 bytes
sea1	19.4	20.4	21.4	22.1	22.7	24.0
sea2	20.8	22.4	23.5	23.9	24.7	26.0
sea3	17.9	19.0	20.1	21.0	21.5	22.6
sea9	23.0	24.2	25.4	26.1	27.4	30.3
sea10	19.6	22.0	23.5	24.1	25.0	26.7
sea11	20.8	22.7	26.4	27.4	29.7	33.3
sea12	19.8	21.6	22.9	23.9	24.9	27.2
sea13	22.3	25.1	26.5	27.4	28.9	31.4
sea14	18.5	18.8	19.4	20.2	20.4	21.3
sea4	16.6	17.4	18.6	20.0	19.8	21.5
sea5	19.4	19.9	20.5	21.3	21.8	23.1
sea6	17.8	19.2	20.2	21.0	21.5	22.7
sea7	19.3	20.3	21.3	21.7	22.2	23.3
sea8	22.1	22.9	23.5	23.9	24.6	25.8
Lena	15.1	16.8	18.8	20.9	21.7	23.2
Church	21.5	22.7	23.8	23.8	24.8	26.4
Olab1	20.1	21.9	23.6	24.4	25.5	27.1
Olab2	19.3	21.3	23.3	24.2	25.4	27.0
Mlab1	21.3	23.2	25.1	25.8	27.1	30.0
Mlab2	20.9	23.2	25.3	25.5	26.8	29.7
Mlab3	21.0	23.2	26.8	27.8	30.0	32.4
Mlab4	20.7	23.1	25.5	26.7	28.6	31.8
Mlab5	23.4	26.0	28.1	29.5	31.9	34.9

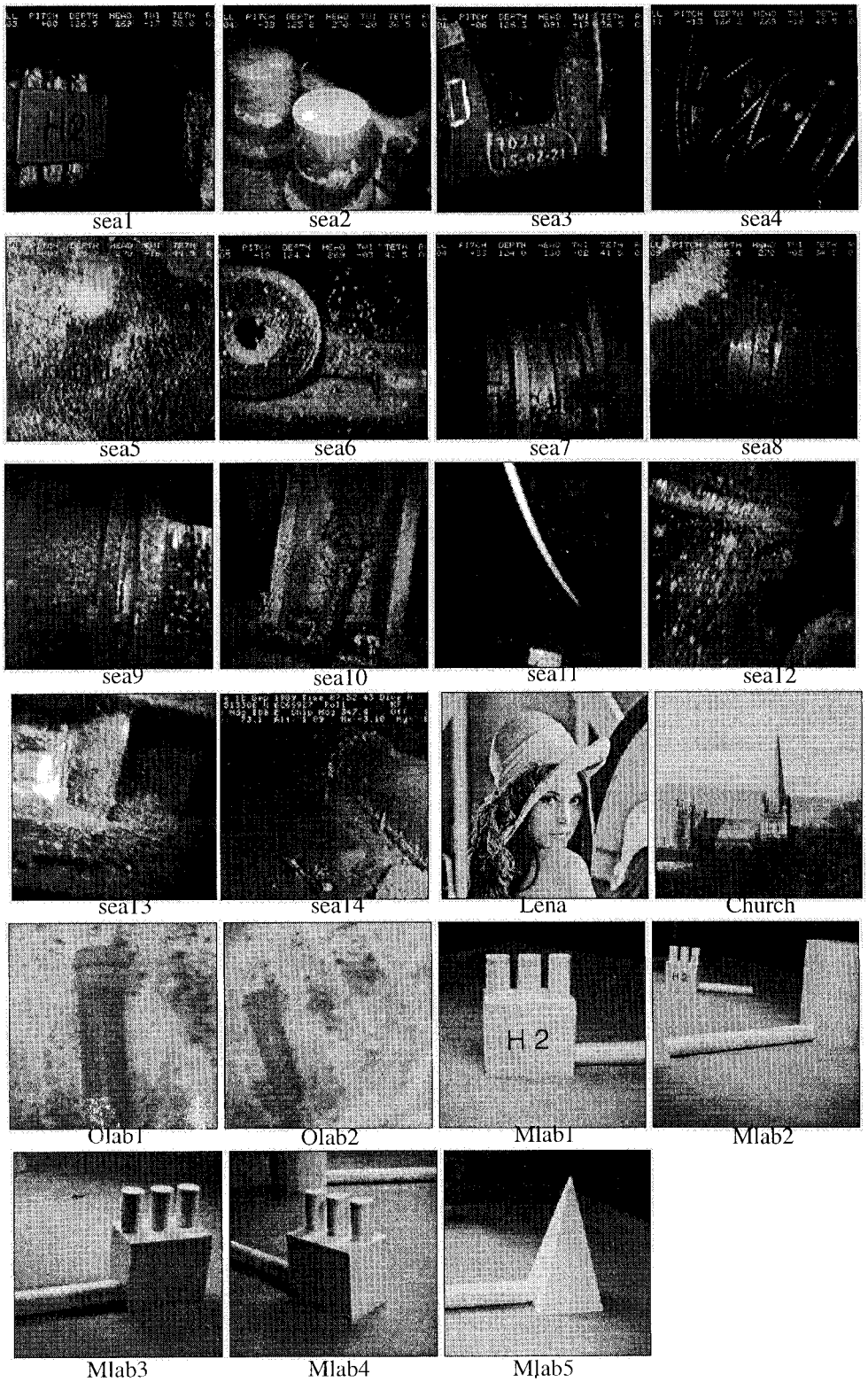


Figure 5. The test images.

Table 2. The SNR values obtained from the JPEG method.

Image	SNR/data size in bytes					
sea1	17.8/542	22.4/949	24.3/1397	26.9/2412	29.3/3970	32.9/7693
sea2	18.8/508	23.3/783	25.3/1134	28.0/2009	30.4/3309	34.2/6334
sea3	17.6/561	21.9/1014	23.7/1487	26.2/2465	28.5/4141	32.0/8293
sea9	18.3/515	23.6/716	26.6/1035	29.4/1941	31.7/3460	34.5/6912
sea10	17.5/552	22.9/847	24.6/1305	26.6/2460	28.5/4850	30.8/9179
sea11	21.9/521	26.2/686	28.3/875	31.2/1459	33.5/2524	36.4/5369
sea12	17.6/532	22.9/887	25.2/1424	27.9/2620	30.4/4542	33.3/8547
sea13	18.2/517	24.6/691	27.0/958	29.7/1769	31.8/3196	34.4/6642
sea14	16.3/574	20.9/1029	22.6/1538	24.4/2623	26.1/4551	28.3/9583
sea4	16.4/588	20.8/1272	22.7/1980	24.8/3094	27.4/5603	30.8/10234
sea5	16.4/529	20.9/1020	22.8/1876	25.4/3665	28.1/6208	31.7/10823
sea6	16.4/548	21.1/1046	22.9/1705	25.3/3140	27.8/5398	31.4/9867
sea7	17.3/542	22.1/907	23.8/1322	26.1/2358	28.5/4183	31.9/8324
sea8	17.5/518	23.3/734	25.2/1036	27.4/1844	29.7/3243	33.3/6543
Lena	16.4/633	21.9/1283	23.8/1979	26.3/3197	28.4/5002	31.5/8736
Church	19.8/506	23.5/687	24.7/953	26.1/1730	27.0/3097	28.9/7863
Olab1	17.5/548	22.9/837	24.7/1250	26.5/2379	28.1/4499	30.0/9209
Olab2	17.7/549	22.7/862	24.5/1302	26.3/2461	27.8/4678	29.7/9205
Mlab1	18.0/532	25.6/729	28.3/892	30.9/1304	32.9/2059	34.7/4223
Mlab2	17.5/528	25.5/722	28.1/876	30.4/1311	31.8/2027	33.3/4049
Mlab3	17.1/523	25.9/689	28.4/871	31.1/1316	32.9/2099	34.4/4313
Mlab4	17.2/537	25.5/770	28.3/977	30.9/1486	32.9/2369	34.7/4783
Mlab5	18.4/513	26.7/661	29.5/807	32.6/1173	34.9/1845	36.7/3586

reconstruction and the signal-to-noise ratio (SNR). The SNR is defined as following [9]:

$$\mathrm{SNR} = 10 \cdot \log_{10}\left(\frac{256^2}{\mathrm{MSE}}\right) \tag{9}$$

where MSE is the mean square error between the original and the reconstructed images.

Two pictorial illustrations of the progressive image reconstruction processes are shown in Fig. 6. Each image sequence contains five intermediate images.

For purposes of comparison, the test images shown in Fig. 5 were also encoded by JPEG. The JPEG software release 3 was used, which was proposed and implemented by the JPEG Group [8]. The experimental results pertaining to JPEG are recorded in Table 2. Two pictorial illustrations of the JPEG images are also shown in Fig. 6.

The performance of JPEG and MOPIC in terms of image data size and SNR are shown graphically in Fig. 7. Based on SNR, the MOPIC algorithm performs better when images are reconstructed from a small amount of data. This meets the MOBATEL requirement. JPEG performs better when images are reconstructed from a large amount of data. This result is expected, since JPEG is designed for high quality image reconstruction.

JPEG and MOPIC exhibit equal (average) SNR when images are reconstructed from approximately 1000 bytes of data. Figure 6 demonstrates clearly that a small amount of data (less than 1000 bytes) can be used to construct useful images. For example, the image reconstructed from 400 bytes of data, which is shown in Fig. 6, can tell a human operator whether a pyramid is in the image and from which direction the pyramid is viewed. Such information is very useful for the navigation of subsea vehicles.

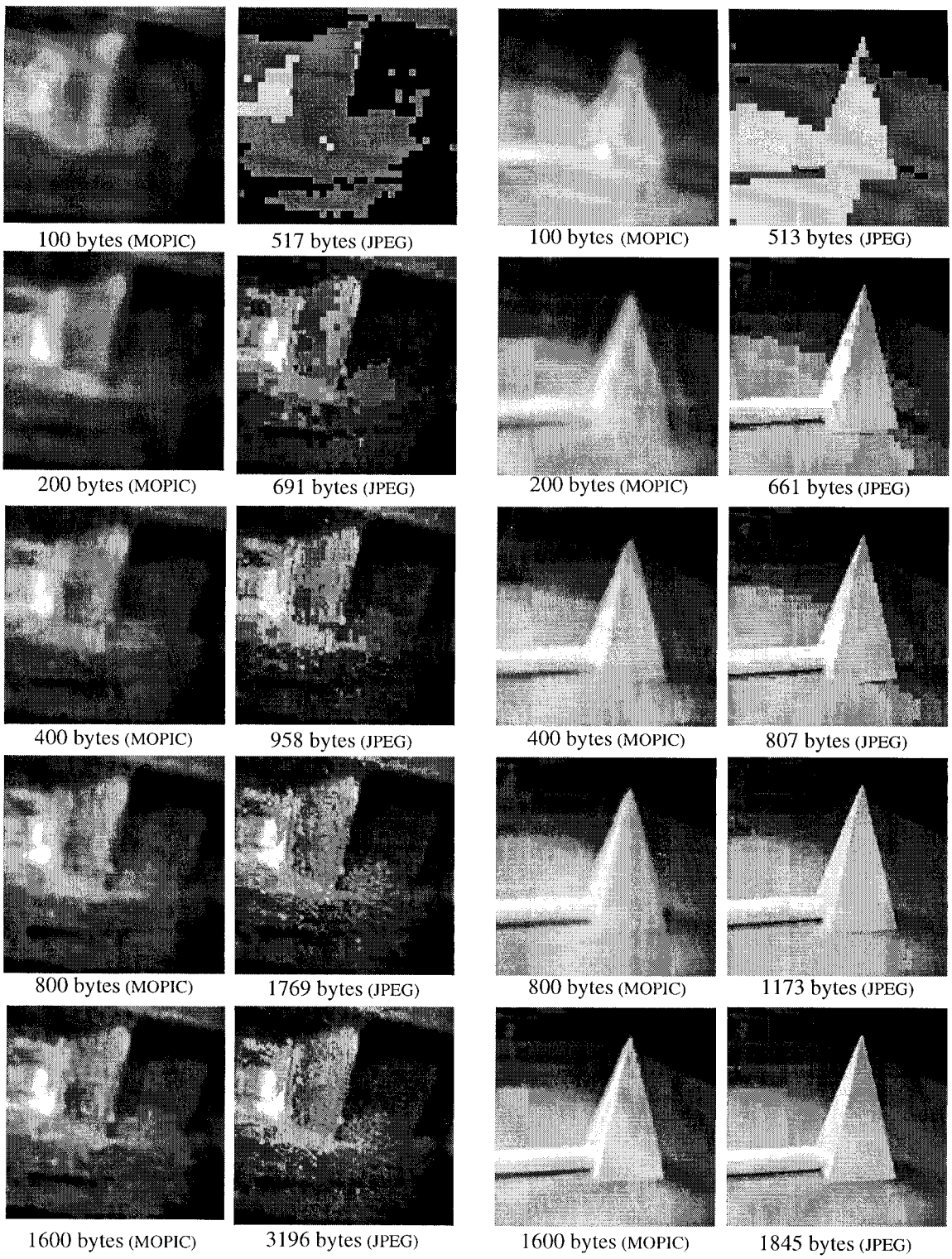


Figure 6. A pictorial comparison of MOPIC and JPEG algorithms.

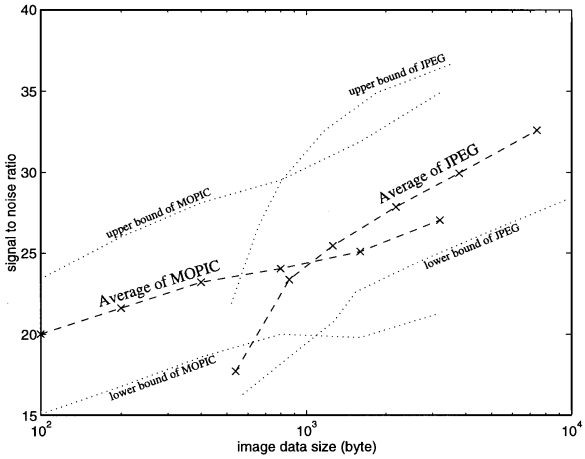


Figure 7. The performances of the JPEG and MOPIC compression algorithms using the images in Fig. 5.

4. Computational Complexity

By considering an image as a single block, MOPIC avoids the blocking effect and reduces the correlation between blocks (refer to Section 2.4), at the cost of increasing the DCT's computational complexity. In MOBATEL, as well as in many robotic systems, computational complexity is key to real-time performance.

In particular, given a 256×256-pixel image transformed by the 2D fast DCT algorithm [9], the computational complexity of the DCT is give by:

$$\mu_M = \frac{3}{4} \cdot N_M^2 \cdot \log_2 N_M \tag{10a}$$

$$\alpha_M = 3 \cdot N_M^2 \cdot \log_2 N_M - 2N_M^2 + 2N_M \tag{10b}$$

when the image is processed as a single block, where μ_M and α_M denote the number of multiplications and additions respectively, and $N_M = 256$ is the block size in MOPIC.

When such an image is divided into $(256/8)^2 = 32^2$ pixel blocks as in JPEG, the computational complexity of the DCT becomes:

$$\mu_J = 32^2 \cdot \left(\frac{3}{4} \cdot N_J^2 \cdot \log_2 N_J \right) \tag{11a}$$

$$\alpha_J = 32^2 \cdot \left(3 \cdot N_J^2 \cdot \log_2 N_J - 2N_J^2 + 2N_J \right) \tag{11b}$$

where μ_J and α_J denote the number of multiplications and additions in JPEG respectively, and $N_J = 8$ is the block size in JPEG.

Examining Eqs. (10) and (11), we can obtain:

$$\frac{\mu_M}{\mu_J} = \frac{\frac{3}{4} \cdot N_M^2 \cdot \log_2 N_M}{32^2 \cdot \left(\frac{3}{4} \cdot N_J^2 \cdot \log_2 N_J \right)} = 2.67 \tag{12a}$$

$$\frac{\alpha_M}{\alpha_J} = \frac{3 \cdot N_M^2 \cdot \log_2 N_M - 2N_M^2 + 2N_M}{32^2 \cdot \left(3 \cdot N_J^2 \cdot \log_2 N_J - 2N_J^2 + 2N_J \right)} = 3.04 \tag{12b}$$

Therefore, compared to JPEG, MOPIC increases the computational complexity of the DCT by 1.67 times and 2.04 times in multiplication and addition respectively. The experiments [9] show that the execution time of DCT is about 500 ms on the HP 730 workstation when N equals to 256. It is feasible to consider a 256×256 pixel image as a single block for image compression, since in most cases, image transmission through an ultrasonic link requires transmission time of 16 to 128 seconds, (corresponding respectively 200 bytes and 1600 bytes of data through an 100 bit/s channel). Consequently, a time delay of several hundred milliseconds is not significant.

5. Transmission Error Effects

In the previous sections, the performance of the MOPIC algorithm was analysed under the assumption that the coded data is accurately transmitted to the decoder. However, in practice, the transmission error effect has to be taken into account, which is especially true in an ultrasonic communication channel.

The most important performance parameter of a digital channel is the bit error rate [4]. Actual bit error rate can range from 10^{-2} to 10^{-8}. The telemetry system to be used in MOBATEL for underwater communication has a bit error rate less than 10^{-6} [10]. Although bit errors in some cases are non-random and clustered in time, in general, a simple model of independent random errors can be used.

If the binary code used in the transmission system is known, such as the *natural binary code, or the folded binary code* [4], a bit error can be converted into channel output error $c(n)$, as follows. Referring to Fig. 8, note that in the presence of transmission error, the reconstructed output signal $y(n)$ will include the effects of both quantization error $q(n)$ and channel error $c(n)$, resulting in a total reconstruction error $r(n)$:

$$r(n) = x(n) - y(n) = q(n) + c(n) \tag{13}$$

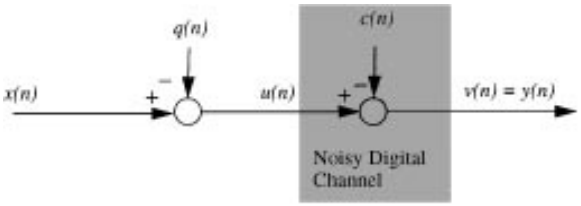


Figure 8. Transmission of quantized amplitudes over noisy channels: $x(n)$, $y(n)$, $q(n)$ and $c(n)$ are input, output, quantization error and channel error respectively. In a noiseless channel, $v(n) = u(n)$.

However, such a quantitative analysis is not always valid in MOPIC, because variable-length entropy coding is used. As mentioned in Section 2.2, symbol-1 in the intermediate sequence is encoded with the VLC, and symbol-2 is encoded with the VLI code. It is important to note that the length of a VLC (Huffman code) is not known until it is decoded, whereas the length of a VLI is stored in its preceding VLC.

When a transmission error occurs in bits that are used to encode symbol-2, the analysis formulated in Eq. (13) is useful. However, a single bit error in bits for encoding symbol-1 can cause loss of codeword synchronization, which leads to a long sequence of erroneous receiver outputs. An example of an image reconstructed with the synchronization problem is shown in Fig. 9. In this case, the erroneous reconstructed image is useless since the synchronization error propagates through out the image and therefore confounds object recognition. In such case, re-transmission would be required.

This synchronization problem can be minimized by proper code selection, which facilitates fast re-synchronization. In some applications, an explicit error protection procedure can be used to deal with channel errors including the synchronization problem. As the expense, the algorithm complexity will be increased and the compression ratio will be reduced.

In MOBATEL, the rate of image re-transmission is important. In particular, the re-transmission rate R_t is a function of channel error rate R_e, average data amount D used for reconstructing each image, and the probability P that the channel error will introduce synchronization problem. In particular:

$$R_t = R_e \cdot D \cdot P \tag{14}$$

In MOBATEL, $R_e = 10^{-6}$(error/bit)[10], $D = 10^4$ (bit/frame) on average, and $P = 0.5$ for MOPIC [11]. Therefore,

$$\begin{aligned} R_t = R_e \cdot D \cdot P &= 10^{-6} \cdot 10^4 \cdot 0.5 \\ &= 0.005(\text{error/frame}) \end{aligned} \tag{15}$$

The above result shows that the synchronization problem, due to channel errors, will typically require one image in 200 to be re-transmitted. Such a rate is acceptable (if not negligible) in an engineering application such as MOBATEL. Note that the transmission of 200 images, where each image contains 10^4 bits of data, will take 2×10^4 second (about 5.5 hours) through a digital communication channel with a bit rate of 100 bits/s.

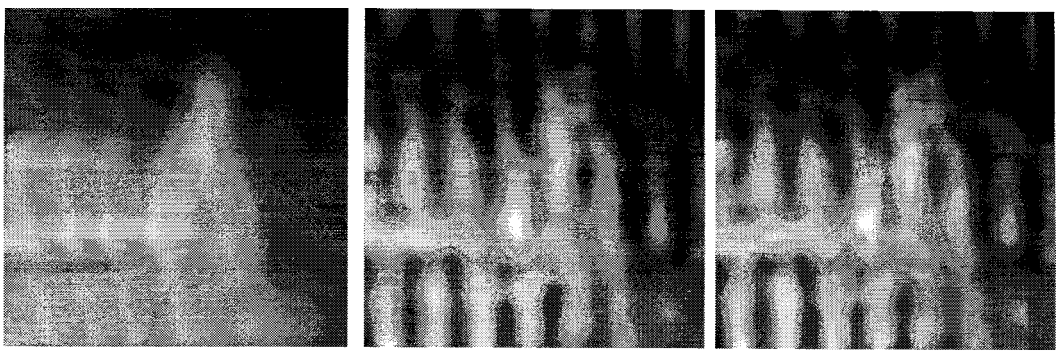


Figure 9. An example of the synchronization problem. The images are reconstructed by: (a) 100 bytes, (b) 200 bytes, and (c) 400 bytes of data respectively. And there is a bit error in the 109th byte of the data stream.

6. Conclusion

The MOPIC (MOBATEL Progressive Image Compression) algorithm proposed in this paper is specially designed for transmitting images through a low rate ultrasonic link. Comparative experiments on real images demonstrate the major advantage of MOPIC over JPEG for its superior signal-to-noise ratio when images are reconstructed from a small amount of data.

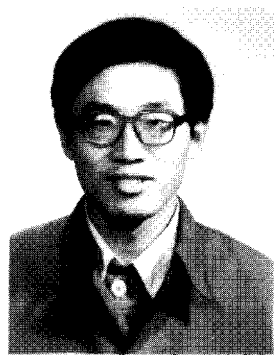
This characteristic is useful in image-based teleoperation over a narrow-band communication link, such as an ultrasonic link with a rate of 100–200 bits/s. MOPIC has been used successfully in subsea telerobotic applications. Example images are provided, with an analysis of synchronization problem due to channel error.

Acknowledgment

The authors would like to thank the reviewer for the very helpful suggestions.

References

1. J.G. Balchen and T.I. Fossen, "Model based teleoperation of an underwater vehicle over a narrow band communication link—MOBATEL," *Proc. of International Advanced Robotics Programme (IARP) 4th Workshop on Underwater Robotics*, Genoa, Italy, Nov. 1992.
2. A.K. Jain, "Image data compression: A review," *Proc. of the IEEE*, Mar. 1981, Vol. 69, No. 3, pp. 349–389.
3. R.C. Gonzalez and R.E. Woods, *Digital Image Processing*, Addison-Wesley Publishing Company Inc., 1992.
4. N.S. Jayant and P. Noll, *Digital Coding of Waveforms, Principles and Applications to Speech and Video*, Prentice-Hall Inc., 1984.
5. Accredited Standards Committee, JPEG Draft Technical Specification (Revision 5), Jan. 1990.
6. K.R. Rao and P. Yip, *Discrete Cosine Transform—Algorithms, Advantages, Applications*, Academic Press Inc., 1990.
7. G.K. Wallace, "The JPEG still picture compression standard," *Communications of the ACM*, pp. 31–43, April 1991.
8. JPEG. *The Independent JPEG Group's JPEG Software*, Release 3 of 17-03-1992, ftp site 137.39.1.9, 192.48.96.9, or 128.214.6.100.
9. C.A. Christopoulos and A.N. Skodras, "Pruning the two-dimensional fast cosine transform," *Proc. the VII European Signal Processing Conf.*, Edinburgh, Scotland, Sept. 13–16, 1994, pp. 596–599.
10. SIMRAD, Product Specifications of SIMRAD HTL 130: Long Range Horizontal Hydroacoustic Telemetry Link, Aug. 1991.
11. S. Yin, *An investigation of Image Capture, Compression and Feature Extraction Algorithm—For an Underwater Telerobotics System*, Dr.ing Thesis, the Norwegian Institute of Technology, 1993.



Shi Yin, B.Sc. East China University of Science, 1982, M.Sc. East China University of Science, 1987, Dr.ing. Norwegian Institute of Technology, 1993. He is currently a research associate at the University of Toronto, Department of Mechanical and Industrial Engineering. His research interests include computer vision, image processing, pattern recognition, emphasis on system performance analysis and their applications.

Jens G. Balchen, Sivilingeniør Norwegian Institute of Technology, 1949, MEng Yale University, 1951, Dr.h.c., Univ. de Liege, 1987. He is professor at the Dept. of Engineering Cybernetics, The Norwegian Institute of Technology, Trondheim, Norway and has been active in control engineering research, consulting and teaching for more than 40 years. He is one of the functions in this organization. His research interests are broad covering modelbased nonlinear control and identification in industrial systems, modelbased teleoperation of underwater vehicles and control theoretic methods applied to ocean biosystems.