PERFORMANCE EVALUATION OF DATA COMPRESSION TRANSFORMS FOR UNDERWATER IMAGING AND OBJECT RECOGNITION

Mark S. Schmalz
mssz@cise.ufl.edu

Gerhard X. Ritter
ritter@cise.ufl.edu

Department of Computer and Information Science and Engineering
University of Florida, Gainesville, FL 32611

Frank M. Caimi
caimi@hboi.edu

Department of Electrical and Software Engineering
Harbor Branch Oceanographic Institution, Ft. Pierce, FL 34946

Abstract -- Underwater (UW) imagery presents several challenging problems for automated target recognition (ATR) using compressed imagery, due to the presence of noise, point-spread function (PSF) effects resulting from camera or media inhomogeneities, as well as loss of contrast and resolution due to in-water scattering and absorption. In practice, sensor noise can severely degrade algorithm performance by producing featural aliasing in the reconstructed (decompressed) imagery.

This paper summarizes recent research in low-distortion, high-rate image compression transforms for ATR applications that require image transmission along low-bandwidth channels such as UW acoustic uplinks. In particular, a novel transform called BLAST has been developed that can achieve compression ratios in the range 50:1 < CR < 280:1 on UW imagery at visually acceptable quality, via simple arithmetic operations over small local neighborhoods. Comparative analysis of performance among BLAST, pyramid coding (EPIC), and visual pattern image coding (VPIC) [9] includes compression ratio, information loss, and computational efficiency measured over a large database of UW imagery. Information loss is discussed in terms of the modulation transfer function (MTF) and several recently-reported image quality measures.

Parallel implementation of the BLAST, VPIC and EPIC transforms is discussed in terms of speed advantages and storage costs. Due to memory requirements, EPIC is generally not suitable for implementation on SIMD-parallel arrays [1]. In contrast, VPIC and BLAST require a maximum of 6kl bytes storage for transformation of each k$x$$l$-pixel encoding block, and are thus suitable for SIMD meshes with small local memories.

I. INTRODUCTION

In previous research, the problem of image compression for transmission along low-bandwidth channels was solved via two techniques. First, a multispectral image is reduced via restriction to a prespecified spectral signature, thereby yielding a Boolean image of probable target locations, which was processed morphologically to remove spurious components [2]. Although this process requires a priori knowledge of the target's approximate signature and geometry, compression ratios ranging from 5,500:1 to 16,500:1 were realized over six-band multispectral imagery of small UW targets in high clutter. In the absence of target information, a second, more general approach compresses monospectral imagery at a mean compression ratio of 0.004 to 0.01 bits per pixel (bpp) using novel transformations.

In 1996, Yin and Balchen [3] reported the development of an enhanced JPEG transformation with 200:1 compression ratio. Although decompressed image quality was sufficient for object segmentation in turbid UW conditions, object surface features remained unresolved. Catapovic et al. [4] have reported the use of MIT's EPIC transform [5] in such applications, with improved image quality. Unfortunately, EPIC requires large local memory and tends to cluster errors in regions of high image variance. Perhaps sufficient for human-in-the-loop viewing, EPIC appears to be inappropriate for ATR applications where representational error must be spatially ergodic. Additionally, Shin et al. [6] have developed a method of compressing underwater sonar images using an adaptive vector quantization technique that continuously updates a VQ codebook to achieve maximum ATR algorithm performance. This is, in a sense, a reverse approach to that taken in Reference 2.

This paper summarizes a novel blockwise method of monospectral image compression based on image and encoding block size, versus the more customary criterion of image content. We show that our method, which involves Blurring, Local Averaging, Sharpening, and

0-7803-4108-2/97/$10.00 ©1997 IEEE

Thresholding filters (hence, the name BLAST), achieves high compression ratios and distributes error uniformly in the spatial sense. Both are key advantages for UW image compression in support of ATR.

In practice, we have measured compression ratios ranging from 50:1 < CR < 280:1 on a diverse UW image database. We have found that BLAST produces visually attractive image reconstructions which can be inspected by humans during target cueing. Mean-squared error (MSE) compares favorably with EPIC's performance at much lower CR and is hence suitable for ATR.

A. Document Organization

Study notation reviewed in Section 1.2 includes *image algebra*, a notation that rigorously unifies linear and nonlinear mathematics in the image domain. Developed at University of Florida under DoD sponsorship since 1984, image algebra preprocessors, interpreters, and class libraries have been implemented on a numerous workstations and parallel processors [7]. Hence, our algorithms are feasible and widely portable.

In Section 2, we discuss the BLAST algorithm and present summary error analyses. Section 3 highlights the performance of EPIC, VPIC, and BLAST on a database of realistic underwater images. Implementational discussion is presented in Section 4, with conclusions and suggestions for future work in Section 5.

B Basic Theory

1. Notation. In the subsequent discussion, the letters **X**, **Y**, and **W** denote point sets, with **F** and **G** denoting value sets. The point sets **X** and **Y** will generally be MxN-pixel arrays and the value set **F** a subset of the real numbers **R** or integers **Z**.

2. Definition. An image is a function $\mathbf{a} : \mathbf{X} \to \mathbf{F}$, written $\mathbf{a} \in \mathbf{F}^{\mathbf{X}}$, that has an alternative graph representation $\mathbf{a} \equiv \{(\mathbf{x}, \mathbf{a}(\mathbf{x})) : \mathbf{a}(\mathbf{x}) \in \mathbf{F}, \mathbf{x} \in \mathbf{X}\}$. We also write $\mathbf{X} = \mathit{domain}(\mathbf{a})$ and $\mathbf{F} = \mathit{range}(\mathbf{a})$.

3. Definition. The *domain compression ratio* of a transform $T : \mathbf{F}^{\mathbf{X}} \to \mathbf{G}^{\mathbf{Y}}$ denoted by CR_{D}, is given by

$$\mathrm{CR}_{\mathrm{D}}(T) = \frac{|\mathbf{X}|}{|\mathbf{Y}|}.$$

4. Definition. The *maximum word width* required to encode values in set S is denoted by *siz*(S).

5. Example. If $\mathrm{S} \subset \mathbf{R}$ and the interval $[\wedge \mathrm{S}, \vee \mathrm{S}]$ is linearly quantized into n bins, then Boolean encoding of S requires $\mathit{siz}(\mathrm{S}) = \log_2(\mathrm{n})$ bits per value in S. In practice, however, proximal values in S may be obscured by such *linear quantization.*

6. Definition. The *range compression ratio* of a transform $T : \mathbf{F}^{\mathbf{X}} \to \mathbf{G}^{\mathbf{Y}}$, denoted by CR_{R}, is given by

$$\mathrm{CR}_{\mathrm{R}}(T) = \frac{\mathit{siz}(\mathbf{F})}{\mathit{siz}(\mathbf{G})}.$$

7. Definition. The *compression ratio* of a transform $T : \mathbf{F}^{\mathbf{X}} \to \mathbf{G}^{\mathbf{Y}}$ is defined as

$$\mathrm{CR}(T) = \frac{|\mathbf{X}| \cdot \mathit{siz}(\mathbf{F})}{|\mathbf{Y}| \cdot \mathit{siz}(\mathbf{G})}.$$

It follows that $\mathrm{CR} = \mathrm{CR}_{\mathrm{D}} \cdot \mathrm{CR}_{\mathrm{R}}$.

8. Definition. A *compressive transform* $T : \mathbf{F}^{\mathbf{X}} \to \mathbf{G}^{\mathbf{Y}}$, has $\mathrm{CR}(T) > 1$, where $|\mathbf{G}| < |\mathbf{F}|$ or $|\mathbf{Y}| < |\mathbf{X}|$ are possible.

9. Definition. The *restriction* of **a** to a subset **W** of **X** is denoted by $\mathbf{a}|_{\mathbf{W}} \equiv \{(\mathbf{x}, \mathbf{a}(\mathbf{x})) : \mathbf{x} \in \mathbf{W}\}$.

The *extension* of image $\mathbf{b} \in \mathbf{F}^{\mathbf{W}}$ to the image $\mathbf{a} \in \mathbf{F}^{\mathbf{X}}$, where $\mathbf{W} \subset \mathbf{X}$, is given by

$$\mathbf{b}|^{\mathbf{a}} \equiv \left\{(\mathbf{x}, \mathbf{c}(\mathbf{x})) : \mathbf{c}(\mathbf{x}) = \begin{cases} \mathbf{a}(\mathbf{x}) \text{ if } \mathbf{x} \in \mathbf{X} \backslash \mathbf{W} \\ \mathbf{b}(\mathbf{x}) \text{ if } \mathbf{x} \in \mathbf{W} \end{cases}\right\}.$$

Similarly, a restriction of $\mathbf{a} \in \mathbf{F}^{\mathbf{X}}$ to a subset of **a** that is characterized by a property S of **F** is given by

$$\mathbf{a}\|_{\mathrm{S}} \equiv \{(\mathbf{x}, \mathbf{a}(\mathbf{x})) : \mathbf{a}(\mathbf{x}) \text{ has property S}, \mathbf{x} \in \mathbf{X}\}.$$

For example, given a threshold $\mathrm{T} \in \mathbf{R}$, if $\mathbf{a} \in \mathbf{R}^{\mathbf{X}}$ then $\mathbf{a}\|_{>\mathrm{T}} \equiv \{(\mathbf{x}, \mathbf{a}(\mathbf{x})) : \mathbf{a}(\mathbf{x}) > \mathrm{T}, \mathbf{x} \in \mathbf{X}\}$.

10. Definition. *Templates* are constructs that map a point to an image. For example, an **F**-*valued template from* **Y** *to* **X**, denoted by the map $\mathbf{t} : \mathbf{Y} \to \mathbf{F}^{\mathbf{X}} \equiv (\mathbf{F}^{\mathbf{X}})^{\mathbf{Y}}$, has *values* denoted by $\mathbf{t}_{\mathbf{y}} \equiv \mathbf{t}(\mathbf{y}) \in \mathbf{F}^{\mathbf{X}}, \mathbf{y} \in \mathbf{Y}$ and *weights* $\mathbf{t}_{\mathbf{y}}(\mathbf{x}) \in \mathbf{F}, \mathbf{x} \in \mathbf{X}$. Thus, we have the graph representation $\mathbf{t}_{\mathbf{y}} \equiv \{(\mathbf{x}, \mathbf{t}_{\mathbf{y}}(\mathbf{x})) : \mathbf{x} \in \mathbf{X}\}$. The point **y** is called the *target point of* **t**, and $(\mathbf{x}, \mathbf{t}_{\mathbf{y}}(\mathbf{x}))$ is called the *target pixel* of **t**.

Given an image $\mathbf{a} \in \mathbf{F}^{\mathbf{X}}$ and the associative, commutative operations $\gamma, \mathrm{O} : \mathbf{F}^{\mathbf{X}} \times \mathbf{F}^{\mathbf{X}} \to \mathbf{F}^{\mathbf{X}}$, the *backward image-template operation*, also called the *right product*, is given by

$$\mathbf{b} = \mathbf{a} \textcircled{?} \mathbf{t} \equiv \left\{(\mathbf{y}, \mathbf{b}(\mathbf{y})) : \mathbf{b}(\mathbf{y}) = \prod_{\mathbf{x} \in \mathbf{X}} (\mathbf{a}(\mathbf{x}) \,\mathrm{O}\, \mathbf{t}_{\mathbf{y}}(\mathbf{x})), \mathbf{y} \in \mathbf{Y}\right\}$$

When $\mathbf{F} \subset \mathbf{C}$, as is customary, the right product can be instantiated as the image-template convolution

$$\mathbf{a} \oplus \mathbf{t} \equiv \left\{(\mathbf{y}, \mathbf{b}(\mathbf{y})) : \mathbf{b}(\mathbf{y}) = \sum_{\mathbf{x} \in \mathbf{X}} (\mathbf{a}(\mathbf{x}) \cdot \mathbf{t}_{\mathbf{y}}(\mathbf{x})), \mathbf{y} \in \mathbf{Y}\right\}$$

by setting $\gamma = +$ and $\mathrm{O} = \cdot$. The *forward image-template operation* or *left product* is symmetrically defined. In either case, $\mathbf{b} \in \mathbf{F}^{\mathbf{Y}}$. Reference 7 contains an exten-

sive discussion of operations between templates, which are similar to image-template operations.

II. BLAST COMPRESSION TRANSFORM

The BLAST transform was discovered by observing that source imagery preprocessed by a sharpening filter generally yields more accurate decompressed representations. We develop this concept mathematically in Sections 2.1-2.2, with analysis in Sections 2.3-2.4.

A. Concept and Basic Theory

BLAST exploits linear systems theory to compress imagery via space-variant quantization of small locally averaged neighborhoods. Decompression reverses the quantization and local averaging process to yield an approximation of the source image.

1. Observation. The advantages of BLAST are (a) more uniform spatial error distribution, with low MSE values possible; (b) local operations only are employed, rendering BLAST efficient on sequential processors and very fast on certain parallel architectures; (c) decompression is efficient, requiring only table lookup and convolution with a small template.

2. Definition. A *tessellation* $\mathbf{Y}$ of an MxN-pixel domain $\mathbf{X}$ is a subset $\mathbf{Y} \subset 2^{\mathbf{X}}$ that satisfies the following properties:

1) $\bigcup_{\mathbf{y} \in \mathbf{Y}} \mathbf{y} = \mathbf{X}$ and

2) $\mathbf{y}_1 \cap \mathbf{y}_2 = \varnothing$ wherever $\mathbf{y}_1 \neq \mathbf{y}_2$, with $\mathbf{y}_1, \mathbf{y}_2 \in \mathbf{Y}$.

3. Definition. Implementationally, an image $\mathbf{a} \in \mathbf{F}^{\mathbf{X}}$ is said to be tessellated by an indexing function $h : \mathbf{X} \rightarrow \mathbf{Y}$ that yields disjoint kx*l*-pixel neighborhoods of $\mathbf{X}$ called *encoding blocks*, where rectangular blocks are customary. For a given $\mathbf{y} \in \mathbf{Y}$, let h have a dual representation $h^* : \mathbf{Y} \rightarrow 2^{\mathbf{X}}$ that returns the domain points in $\mathbf{X}$ of the y-th encoding block, such that $\mathbf{x} \in h^*(h(\mathbf{x}))$. A set of *landmarks* for $\mathbf{Y}$ is a set of points of $\mathbf{X}$ such that for each $\mathbf{y} \in \mathbf{Y}$ there exists a unique $\mathbf{x} \in \mathbf{X}$ such that $\mathbf{x} \in \mathbf{Y}$. In practice, a landmark such as the block centroid could be located by *f*.

B. Algorithm

Let a source image $\mathbf{a} \in \mathbf{R}^{\mathbf{X}}$ be tessellated into kx*l*-pixel blocks $\mathbf{b}(\mathbf{y})$, $\mathbf{y} \in \mathbf{Y}$, per Definition 2.1.3 and Figure 1. Given a template $\mathbf{s} \in (\mathbf{R}^{\mathbf{X}})^{\mathbf{X}}$ that could be space-variant, and an operation $f_{\mathrm{L}} : \mathbf{R}^{\mathbf{X}} \rightarrow (\mathbf{Z}_{2^m})^{\mathbf{Y}}$, BLAST *compression* is expressed as

$$\mathbf{a}_c = f_{\mathrm{L}}(\mathbf{a} \oplus \mathbf{s}) \qquad \text{(I)}$$

such that, given a quantization operation $q : \mathbf{R} \rightarrow \mathbf{Z}_{2^m}$ having an approximation $q^* : \mathbf{Z}_{2^m} \rightarrow \mathbf{R}$ to its inverse, the y-th encoding block $\mathbf{b}(\mathbf{y})$ is locally averaged and quantized as

$$f_{\mathrm{L}}(\mathbf{b}(\mathbf{y})) = q\left(\frac{\Sigma \mathbf{b}}{kl}\right).$$

Given a variant template $\mathbf{t} \in (\mathbf{R}^{\mathbf{X}})^{\mathbf{X}}$ and an operation $f_{\mathrm{B}} : (\mathbf{Z}_{2^m})^{\mathbf{Y}} \rightarrow \mathbf{R}^{\mathbf{X}}$, *decompression* partially reverses the effect of the projection $f_{\mathrm{B}}(f_{\mathrm{L}}(\mathbf{a}))$ and convolution with s, approximating the source image as

$$\mathbf{a} \approx f_{\mathrm{B}}(\mathbf{a}_c) \oplus \mathbf{t}. \qquad \text{(II)}$$

Here, f_{B} assigns the value $q^*(\mathbf{a}_c(\mathbf{y}))$ to each point in $h^*(\mathbf{y})$ to yield an image on $\mathbf{X}$, where h^* was defined in Section 2.1.3.

C. Complexity

Given appropriate configuration of s and t, which for two-dimensional imagery are customarily sparse templates no larger than 7x7 pixels, only local operations over small neighborhoods (e.g., no larger than kx*l* pixels) are required to implement BLAST compression or decompression. Assuming that (a) $\mathbf{X}$ is an MxN-pixel array, (b) the mean of $|S(\mathbf{s}_y)|, \mathbf{y} \in \mathbf{X}$ (or $|S(\mathbf{t}_y)|, \mathbf{y} \in \mathbf{X}$) is denoted as p (q), and (c) encoding

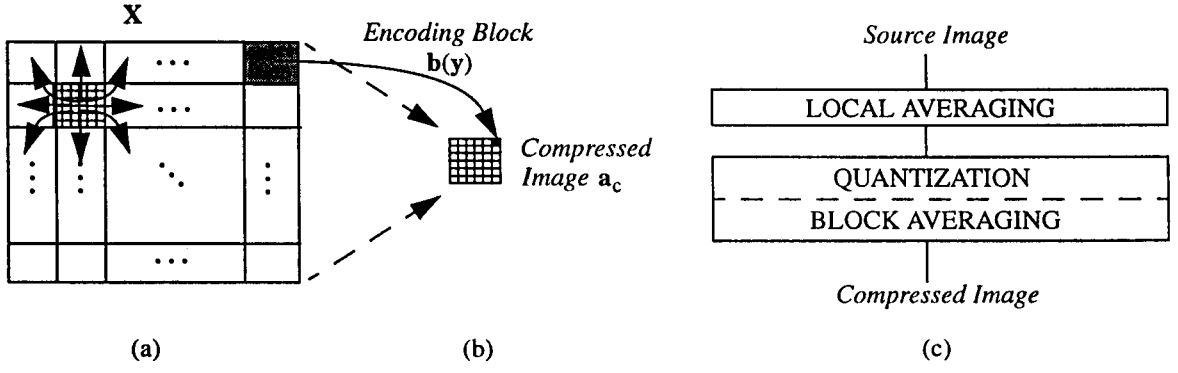

Figure 1. BLAST compression (a) blockwise tessellation and local averaging (combination shown schematically by arrows), (b) compressed image containing mean of block $\mathbf{b}(\mathbf{y})$, and (c) algorithm block diagram.

blocksize is k×l pixels, the associated computational cost is itemized in Table 1.

The approximate cost of compression is thus given by $W_c(\mathbf{X}, k, l, p) \approx |\mathbf{X}| \cdot (kl + p + 4)$ multiplications, where approximately $W_d(\mathbf{X}, q) = |\mathbf{X}| \cdot q$ multiplications are required for decompression.

For example, given $CR_D \approx kl$ and typical values of $9 \le p, q \le 13$, the compression and decompression costs are given by $W_c \propto CR_D + p$ and $W_d \propto q$. Assuming a typical range of $50{:}1 \le CR_D \le 100{:}1$, image compression requires from at most from 60 to 115 multiplications per pixel. In contrast, decompression requires approximately 10 to 15 multiplications per pixel, yielding a total cost of 70 to 130 multiplications per pixel.

D. Information Loss

Given Equations (I) and (II), BLAST can be expressed in algebraic form, as follows. For simplicity, let source image $\mathbf{a} = (a_1, a_2, \ldots, a_{|\mathbf{X}|})$ and templates $\mathbf{s} = (s_{-1}, s_0, s_1)$ and $\mathbf{t} = (t_{-1}, t_0, t_1)$, where the target pixel is italicized. Expansion of Equations (I) and (II) in such terms yields the following system of equations:

$$
\begin{array}{lll|l|l|l}
\mathbf{a} & = (& a_1\,, & a_2\,, & a_3\,, & a_4 \quad \ldots\ ) \\
\mathbf{a} \oplus \mathbf{s} & = (& a_1 s_0 + a_2 s_1\,, & a_1 s_{-1} + a_2 s_0 + a_3 s_1\,, & a_2 s_{-1} + a_3 s_0 + a_4 s_0\,, & a_3 s_{-1} + a_4 s_0 + a_5 s_0, \ldots\ ) \\
\mathbf{a}_c = f_L(\mathbf{a} \oplus \mathbf{s}) & = (& \boxed{q[a_1(s_0 + s_{-1}) + a_2(s_0 + s_1) + a_3 s_1]}\,, & & \boxed{q[a_2 s_{-1} + a_3(s_{-1} + s_0) + a_2(s_0 + s_1) + a_5 s_2]}\,, & \ldots\ ) \quad \text{(III)} \\
f_B(\mathbf{a}_c) & = (& q^*(\mathbf{a}_c(1))\,, & q^*(\mathbf{a}_c(1))\,, & q^*(\mathbf{a}_c(2))\,, & q^*(\mathbf{a}_c(2))\,, \quad \ldots\ ) \\
f_B(\mathbf{a}_c) \oplus \mathbf{t} & = (& q^*\mathbf{a}_c(1) \cdot (t_0 + t_1) & q^*(\mathbf{a}_c(1)) \cdot (t_{-1} + t_0) + q^*(\mathbf{a}_c(2)) \cdot t_1\,, & q^*(\mathbf{a}_c(1)) \cdot t_{-1} + q^*(\mathbf{a}_c(2)) \cdot (t_0 + t_1) & q^*(\mathbf{a}_c(2)) \cdot (t_{-1} + t_0) + q^*(\mathbf{a}_c(3)) \cdot t_1\,, \ldots\ )
\end{array}
$$

where vertical lines separate source domain points and boxes delimit compressed image values. For example, the compressed image value $\mathbf{a}_c(1)$, derived from the expressions $(\mathbf{a} \oplus \mathbf{s})(1)$ and $(\mathbf{a} \oplus \mathbf{s})(2)$, gives rise to the decompressed image pixel values $f_B(\mathbf{a}_c)(1)$ and $f_B(\mathbf{a}_c)(2)$. It is easily verified that the preceding system of equations can be solved in linear form via well-known approximation methods, to yield $\mathbf{t}$ given $\mathbf{s}$, or vice versa.

Given $\mathbf{a}_c$, $\mathbf{s}$, and $\mathbf{t}$, solution of Equation (III) for non-boundary pixel values (e.g., domain points 3 and 4) yields the following reconstruction, which is a linear approximation of a source pixel value a_3:

$$
\begin{aligned}
a_3 &\approx q^*(\mathbf{a}_c(1) \cdot t_{-1} + \mathbf{a}_c(2) \cdot (t_0 + t_1)) \\
&= q^*(q(a_1 \cdot (s_{-1} + s_0) + a_2 \cdot (s_0 + s_1) + a_3 \cdot s_1) \cdot t_1 \\
&\quad + q(a_2 \cdot s_{-1} + (a_3 + a_4) \cdot (s_{-1} + s_0) + a_5 \cdot s_2) \cdot (t_0 + t_1))\,.
\end{aligned}
$$

Assuming, for purposes of simplicity that $r \in range(\mathbf{a})$ and $q^*(q(r))$ has error ε_q, the longest computational path in the dataflow graph of Equation (X), derived from the argument of q^*, is given in functional notation as

$$(+\ (\times\ (\Sigma\ (\times\ (+\ s_{-1}\ s_0)\ a_1)\ (\times\ (+\ s_0\ s_1)\ a_2)\ \ldots)\ (+\ t_0\ t_1)))\,.$$

Per Reference [-], the corresponding error function is obtained by relabelling the nodes of Equation (XI) with their respective error functions.

Table 1. Cost analysis of the BLAST compression transform (per Equations I and II).

Task	Operation	Cost
Compression	$\mathbf{a} \oplus \mathbf{s}$	average cost of MNp multiplications and MN(p-1) additions
	f_L	$kl \cdot H$ mult's and $k(l-1) \cdot H$ additions for local averaging, where $H = \lceil |p_1(\mathbf{X})|/k \rceil \cdot \lceil |p_2(\mathbf{X})|/l \rceil$
	q	$4 \cdot |\mathbf{X}|$ multiplications and $3 \cdot |\mathbf{X}|$ additions for quantization
Decompression	f_B	$2 \cdot |\mathbf{X}|$ I/O operations for mean projection to $\mathbf{X}$
	$f_B(\mathbf{a}_c) \oplus \mathbf{t}$	average cost of MNq multiplications and MN(q-1) additions

That is, if $p_{ij} = a_i \cdot (s_j + s_{j+1})$, $r_{ij} = a_i \cdot s_j$, $\varepsilon_p \equiv (\varepsilon_x\ (\varepsilon_+\ \varepsilon_s\ \varepsilon_s)\ \varepsilon_a)$, and $\varepsilon_r \equiv (\varepsilon_x\ \varepsilon_s\ \varepsilon_a)$, then the resultant error function is given by

$$\varepsilon_{a(3)} = ((+\ \varepsilon_q\ \varepsilon_+(\varepsilon_x(\varepsilon_\Sigma\ \varepsilon_p\ \varepsilon_p\ \varepsilon_r)\ \varepsilon_t) \\ (\varepsilon_x(\varepsilon_\Sigma\ \varepsilon_r\ \varepsilon_p\ \varepsilon_p\ \varepsilon_r)\ (\varepsilon_+\ \varepsilon_t\ \varepsilon_t)))) .$$

This reduces to the following general form for $\varepsilon_{\tilde{a}}$, the error function corresponding to non-boundary pixels in the approximation $\tilde{\mathbf{a}}$ of $\mathbf{a}$:

$$|\varepsilon_{\tilde{a}}| \le \varepsilon_q + \left(\frac{|2\varepsilon_p| + |\varepsilon_r|}{v} + \frac{|\varepsilon_t|}{t_1}\right) \cdot v\varepsilon_t \\ + \left(\frac{|2\varepsilon_p| + |2\varepsilon_r|}{w} + \frac{|2\varepsilon_t|}{t_0 + t_1}\right) + w(t_0 + t_1) ,$$

where the coefficients $v = p_{1,-1} + p_{2,0} + r_{3,1}$ and $w = r_{2,-1} + p_{3,-1} + p_{4,-1} + r_{5,2}$.

E. Preliminary Results

The BLAST algorithm was initially tested on a small database of ten multispectral images (six spectral bands, of which three exhibited sufficient contrast), as well as public-domain imagery such as *lena*. Results for an example UW image and the *lena* image are shown at compression ratios of 32:1, 98:1, and 200:1, where rectangular encoding blocks were employed. The use of overlapping elliptical encoding blocks, each subsampled near its periphery, increased the visual attractiveness of BLAST's output as shown in Figure 3.

Elliptical encoding blocks produce several useful effects. First, the disruption of oblique straight lines with increasing blocksize, a well-known deficiency of rectangular-block tessellation, is reduced by BLAST's use of curvilinear block boundaries. This yields a more visually attractive rippled edge, versus a staircase effect. Examples are given in Figure 3c-d, on the edges of Lena's hat and oblique boundaries near the mirror.

Second, overlapping blocks tend to distribute more information near the periphery of an encoding block to its neighboring blocks. This decreases entropy via averaging, which further enhances compressibility. Third, sparse sampling at the block perimeter (versus dense sampling at its center) can have the effect of spatially interleaving encoding block information during decompression. This further reduces apparent disruption of oblique lines without significantly blurring boundaries.

Such effects are visible in Figure 3d, where the domain compression ratio CR_D = 121:1, due to the use of 121-pixel encoding blocks. Since CR = 242:1, this implies that the range compression ratio CR_R = 2:1, which is achieved via the quantization inherent in f_L, as illustrated in Equation (I).

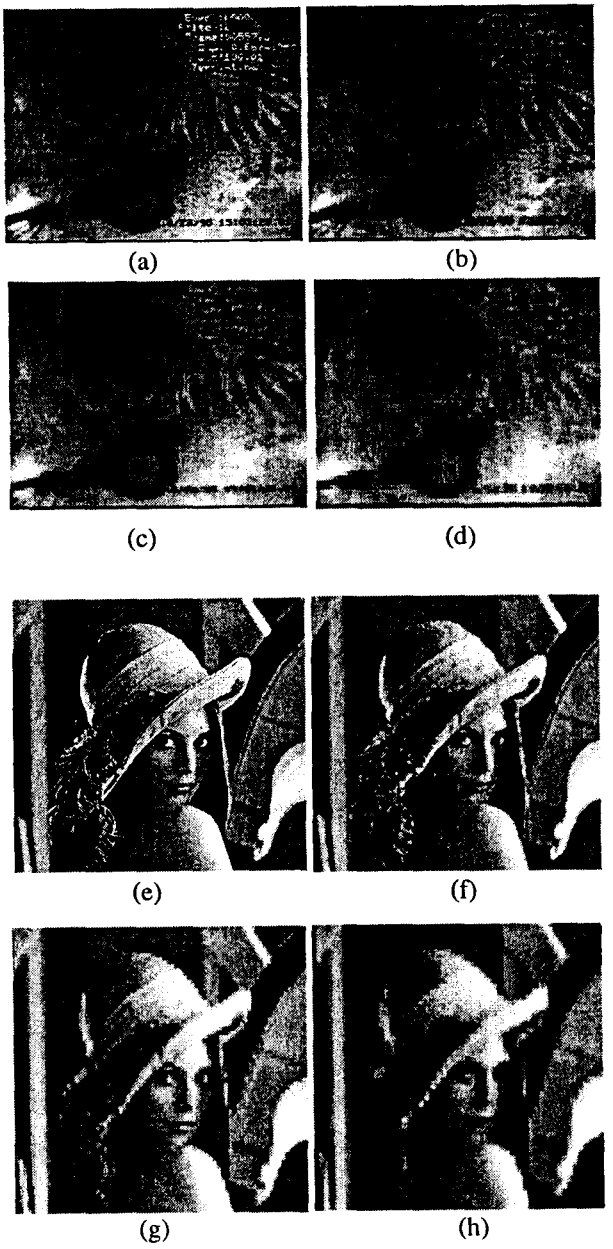

Figure 2. Example BLAST output: (a,e) source images; (b,f) CR = 32:1, (c,g) CR = 98:1, and (d,h) CR = 200:1.

Depending upon the quantization scheme (e.g., linear or histogram-based) and image statistics, range compression ratios ranging from 1.2:1 to 2.67:1 can be achieved with visual clarity. At the upper range limit, this implies a 3 bpp greyscale . We are currently investigating Huffman coding of greyscale information, to further reduce entropy and obtain range compression ratios approaching 3:1. Theory presented in Section 2 suggests that image compression ratios approaching 300:1 could be achieved with this method. Early results are promising and will be analyzed in future research.

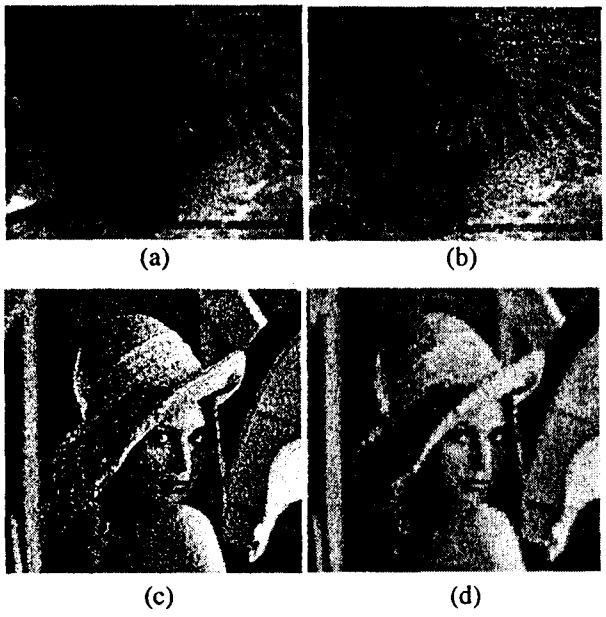

Figure 3. BLAST with subsampled elliptical encoding blocks using Figure 2 a,e as source images: (a,c) CR = 98:1; (b,d) CR = 242:1.

III. COMPARATIVE ANALYSIS

The utility of various compressive transforms for UW applications was ascertained by applying VPIC, BLAST, and EPIC to a database of realistic UW images, using published image quality measures (IQMs) as primary performance criteria. By using a suite of IQMs rather than one measure such as MSE, greater breadth and robustness of comparison was achieved. Additional performance measures included work requirement, sequential computation time, and space requirement.

A. Image Database

Test imagery comprised a database of 15 multispectral images comprised of 6 spectral bands of 720x480 pixels resolution. Each image depicted scenes containing coral, sand, rocks, or vegetation, and 10 images contained a target of interest. Imagery was acquired with a Xybion IMC-301 intensified CCD camera, where the combined point-spread function (PSF) of camera and water column exhibited a FWHM of 7.4 pixels [2].

B. Performance Measures

Previous investigators have extensively discussed the paucity of useful image quality measures (IQMs), which is due in part to lack of knowledge concerning human visual system (HVS) function. Eskicioglu [8] summarized and evaluated published IQMs with applications in performance analysis of image compressors. Due to lack of knowledge about HVS function, Eskicioglu's work cannot be rigorously associated with HVS parameters and, hence, more objective measures of image appearance.

In contrast, we have identified primary degradations in compressed UW imagery that adversely influence ATR, and have adapted published IQMs accordingly. Given a source image **a** and the corresponding decompressed image **b**, our performance measures include:

1) *Compression Ratio*, required for bandwidth reduction and computed per Definition 1.2.7.
2) *MSE*, a customary constraint on ATR algorithm design, given by $\mathrm{MSE} = \|\mathbf{a}-\mathbf{b}\| / \sqrt{|domain(\mathbf{a})|}$;
3) *MTF* - key to computing high-frequency cutoff f for feature visibility. The MTF is computed from the Fourier transform $\mathcal{F}(\mathbf{a})$.
4) *Blocking Effect*, obtained from the MTF as $\Sigma[\mathcal{F}(\mathbf{a})|_f]$, where f corresponds to blocksize;
5) *Edge Integrity* assesses boundary distortions by computing the linearity of a straight boundary in a neighborhood of **a** using linear regression;
6) *Work Requirement*, which is currently predicted from theory and measured as execution time on a variety of sequential workstations that include CISC personal computers and RISC machines; and
7) *Space Requirement*, expressed as bytes of local memory required for transform computation.

The automatic determination of edge integrity is currently difficult, since boundaries must be isolated manually. As a result, edge integrity can be tested only sporadically. However, we are currently developing a line-finding algorithm based on the Hough transform that will locate and analyze linear objects automatically.

C. Experimental Results

The VPIC, EPIC, and BLAST transforms were applied to the image database discussed in Section 3.1, with performance measures given in Section 3.4. Figure 4 depicts MSE and blocking effect as a function of CR. Note that BLAST's MSE compares favorably with EPIC, but VPIC is not useful when CR > 70 due to the featural aliasing effects of large (8x8-pixel) exemplars. In contrast to codebook-based transforms such as VPIC, which is data-dependent, BLAST and EPIC better preserve image quality at high compression ratios.

Figure 4c illustrates the degradation of edge integrity with compression ratio. Note that this varies approximately inversely with the blocking effect shown in Figure 4b. In contrast, when elliptical subsampled blocks are employed in BLAST, both the blocking effect and edge degradation are drastically reduced. In its current formulation, EPIC, which is based upon the topology of rectangular blocks, cannot effectively exploit such modifications.

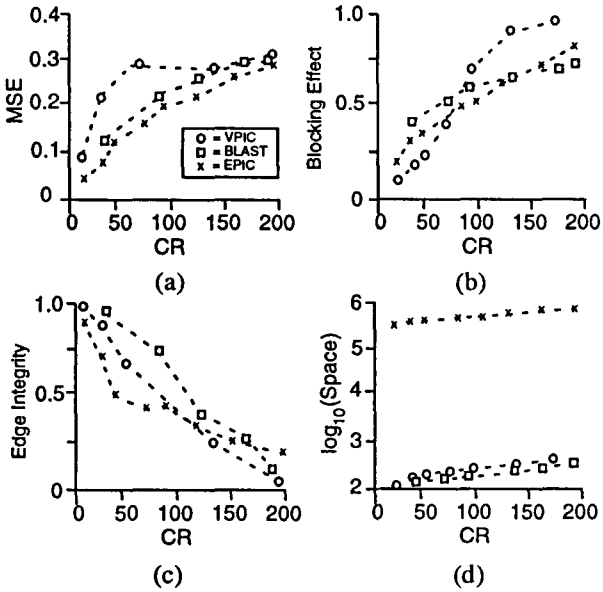

Figure 4. Compression transform performance vs. compression ratio: (a) MSE; with normalized (b) blocking effect; (c) edge integrity; and (d) space requirement.

IV. IMPLEMENTATIONAL DISCUSSION

Figure 4d illustrates the dependence of space complexity (local memory) and execution time for VPIC, EPIC, and BLAST. Note that EPIC requires that the entire image be stored in local memory, while VPIC and BLAST require local transformation of one encoding block only. Given that VPIC and BLAST each require an average of six registers per pixel for storage of intermediate results, parallel implementation of VPIC and BLAST on SIMD meshes is feasible.

In particular, given kx*l*-pixel encoding blocks, a total of 6k*l* bytes is required. Since k,*l* = 10 is currently the limiting case, this implies local storage of 600 bytes, which is achievable in the majority of commercially available SIMD meshes.

We are currently designing a parallel implementation of VPIC and BLAST on Lockheed-Martin's Parallel Algebraic Logic (PAL) processor. PAL-I is a SIMD-parallel image processor that achieves 385 MOPS per board and is linearly scalable to 32 boards. PAL-II, which is currently being prototyped, is designed to achieve 2.5 GFLOPS (32-bit floating point multiplications/sec) per board. The PAL series of processors are programmable in Image Algebra C++ [7], which implements the subset of image algebra given in this paper.

V. CONCLUSIONS

The compression of underwater imagery for transmission along low-bandwidth acoustic channels requires high image quality as well as low bit rate, if ATR is to be conducted over the decompressed imagery. This paper reviews two well-known image compression transforms, VPIC and EPIC, and introduces a third, novel compression technique called BLAST. Although VPIC is capable of moderate image compression ratios (e.g., less than 60:1 with visually acceptable quality), it suffers from space-variant error and tends to obscure fine featural detail at scales less than 0.25 of the blocksize. EPIC has similar shortcomings at high resolution, and tends to cluster errors in regions of high image variance. Additionally, EPIC requires a large memory model that is infeasible for parallel implementation on SIMD meshes with small local memory.

In contrast, BLAST achieves high compression (CR in excess of 250:1) with visually acceptable image quality, spatially uniform error, and predictable spatial distortions. As a key advantage, the BLAST transform uses simple arithmetic operations over small neighborhoods only, which facilitates parallel implementation.

ACKNOWLEDGEMENT

The authors gratefully acknowledge support (under Grant N00014-96-1-5019), and efforts on behalf of Dr. T. Curtin, of the Office of Naval Research.

REFERENCES

[1] Jackson, R., P.C. Coffield, and J.N. Wilson. "New SIMD computer vision architecture with image algebra programming environment", *IEEE Aerospace Applications Conference Proceedings* 1:169-185 (1997).

[2] Schmalz, M.S. , G.X. Ritter, and F.M. Caimi. "Data compression techniques for underwater imagery", *Proceedings IEEE OCEANS '96 Conference* 2:929-936 (1996).

[3] Yin, S. and J.G. Balchen. "Image compression and transmission through a low-rate ultrasonic link in subsea telerobotic applications", *Journal of Mathematical Imaging and Vision* 7:41-52 (1997).

[4] Freitag, L.E., J.A. Catapovic, and R.L. Eastwood. "Acoustic communications system for the AMMT program", *Oceans Conference Record (IEEE)* Supplement: 87-92 (1996).

[5] Adelson, E.H. and E.P. Simoncelli. "Subband image coding with three-tap pyramids", in *Proceedings of the 1990 Picture Coding Symposium* (1990).

[6] Shin, F.B., D.H. Kil, and G.J. Dobeck. "Impact of lossy image compression on automatic target recognition performance", *Proceedings IEEE OCEANS '96 Conference* 2:943-948 (1996).

[7] Ritter, G.X. and J.N. Wilson. *Handbook of Computer Vision Algorithms in Image Algebra*, Boca Raton, FL: CRC Press (1996).

[8] Eskicioglu, A.M. "Image quality measures and their performance", *IEEE Transactions on Communications* 43:2959-2965 (1995).

[9] Chen, D. and A.C. Bovik. "Visual pattern image coding", *IEEE Transactions on Communications* 38:2137-2146 (1990).

[10] Curtin, T., J. Bellingham, J. Catapovic, and D. Webb. "Autonomous oceanographic sampling networks", *Oceanography* 6:3 (1993).