Integer Cosine Transform for Image Compression

K.-M. Cheung, F. Pollara, and M. Shahshahani
Communications Systems Research Section

This article describes a recently introduced transform algorithm called the integer cosine transform (ICT), which is used in transform-based data compression schemes. The ICT algorithm requires only integer operations on small integers and at the same time gives a rate-distortion performance comparable to that offered by the floating-point discrete cosine transform (DCT). The article addresses the issue of implementation complexity, which is of prime concern for source coding applications of interest in deep-space communications. Complexity reduction in the transform stage of the compression scheme is particularly relevant, since this stage accounts for most (typically over 80 percent) of the computational load.

I. Introduction

The rate-distortion performance of three transform-based coding schemes used to compress the test images for the Comet Rendezvous Asteroid Flyby (CRAF)/Cassini Project was presented in [1]. More recently, the issue of implementation complexity, which is of prime concern to spacecraft applications, was addressed. The computational bottleneck of transform-based algorithms lies in the front-end transform stage, which accounts for over 80 percent of the computational load of these compression schemes. This article describes a recently introduced transform algorithm called the integer cosine transform (ICT), which requires only integer operations on small integers and at the same time has rate-distortion comparable to that of the floating-point discrete cosine transform (DCT), which is the most practical and near optimal approach known for data compression. The implementation complexity of the ICT is substantially lower than that of the DCT, and is comparable to that of the Hadamard transform (HT). The ICT is a practical approach to achieving the high-rate deep-space communications that are possible with the DCT.

II. Background: Transform-Based Schemes

In preparing the test images for the CRAF/Cassini Project, three transform-based encoding algorithms were used to compress a set of seven planetary images [1], which are continuous-tone gray-scale, with pixel values ranging from 0 (black) to 255 (white). All three algorithms can be viewed as consisting of three stages, as illustrated in Fig. 1: the data transform stage, the quantization stage, and the entropy-coding stage. The compression algorithms work on a block-by-block basis, i.e., they compress an 8×8 block of the picture at a time. In each algorithm, the encoder first applies an 8×8 floating-point DCT or an 8×8 HT to the picture block to generate an 8×8 block of transform coefficients. These numbers are then quantized by a predetermined 8×8 quantization template to integer values. Most quantized values have small magnitudes. Due to the skewed distribution of the quantized transform coefficients, compression is achieved by assigning shorter transmission-bit patterns to the more frequently occurring integers. This is realized in the last stage of the compression scheme, the entropy coder, which maps the quantized values to appropriate transmission-bit patterns.

In the CRAF/Cassini data compression experiment, all three transform-based schemes used the same DCT or HT (stage 1) and the same quantization template (stage 2). The difference lies in the choice of entropy coder in the third stage, where one may use the Joint Photographic Expert Group (JPEG) Huffman code [2,12], an arithmetic code [2], or the Gallager–van Voorhis–Huffman (GVH) code [3]. In general, the DCT-based schemes are more effective (0.1 to 0.3 bits per pixel) than the HT-based schemes, especially in the high bit rate (near lossless) range. However, the more effective DCT-based schemes are more computationally intensive than the HT-based schemes. The major computational burden of DCT-based schemes lies in the DCT stage, which requires a large number of floating-point multiplications and additions. HT-based schemes, on the other hand, require only integer additions and subtractions in the transform stage. From the hardware's point of view, floating-point operations are much slower and more difficult to implement than the corresponding integer operations. For a general N-point DCT, a straightforward algorithm [4] that yields a simple regular implementation and a small chip size requires $2N^3$ multiplications and $2N^3$ additions. A more sophisticated N-point fast DCT [5], where N has a power of 2, that uses complex data-shuffling strategies still requires $N^2 \log_2 N$ multiplications and $N(3N \log_2 N - N + 1)$ additions. The large number of floating-point operations required to perform DCT, particularly for large N, is the computational bottleneck for all DCT-based signal-processing schemes.

III. Integer Cosine Transform

Recently Choy, Cham, and Lee [6] proposed a new 8-point transform called the integer cosine transform (ICT), which requires only integer multiplications and additions, and thus is much simpler to implement than the DCT. An ICT chip was fabricated and was proven to be efficient in both silicon area and speed [6]. The 8×8 ICT matrix suggested in [6] is given in Fig. 2(c). Notice that the elements in the matrix are all integers, and the ICT matrix B in Fig. 2(b) has sign and magnitude patterns that resemble those of the DCT matrix A in Fig. 2(a). The similarity of the ICT matrix to the DCT matrix, together with the orthogonality property of the ICT ($BB^t = \Delta$, where Δ is a diagonal matrix), guarantees that the ICT, as well as its inverse, possesses the same transform structure as the DCT. Thus, any fast DCT algorithms can be used to compute a fast ICT.

This 8×8 ICT matrix was used to compute a two-dimensional 8×8 transform and then compress the planetary images *saturn1* and *saturn2*. The transform coefficients were quantized by using the same quantization template as in the aforementioned DCT-based schemes. The entropy of the quantized transform coefficients and the mean square error (MSE) of the reconstructed picture were computed, and the results are shown in Fig. 3. These simulation results indicate that any difference in rate-distortion performance resulting from using the floating-point DCT or the ICT is unnoticeable.

Although the 8-point ICT proposed by Choy, Cham, and Lee performs remarkably well, it is quite ad hoc, and no general mathematical formulation of the ICT is given in [6]. The contributions of this article are to put the ICT into a more formal mathematical setting, and to generalize their idea to any N-point ICT. The mathematical properties of the ICT are investigated in the following sections. Since the ICT is separable, and the extension of the one-dimensional ICT to two dimensions is straightforward, this article focuses on the one-dimensional case. Section IV gives a characterization of ICT matrices. An 8×8 ICT matrix that is multiplication-free and requires only binary additions and shifts is given in Section V (the MSE versus entropy performance of the multiplication-free ICT, the original ICT of [6], and that of the floating-point DCT are shown in Fig. 3). A general procedure for the construction of an $N \times N$ ICT matrix is obtained in Section VI; and two 16×16 ICT matrices, one with only small integer entries and one with all entries' powers of two (multiplication-free) are exhibited in Section VII.

IV. Mathematical Properties of the ICT

The integer cosine transform and the discrete cosine transform are closely related. Let C and A be the respective ICT and DCT $N \times N$ matrices. An orthonormal matrix (i.e., $AA^t = I$), $A = [a_{kn}]$, is defined as follows for $0 \le n \le N-1$:

$$a_{kn} = \frac{1}{\sqrt{N}} \qquad k = 0$$

$$= \sqrt{\frac{2}{N}} \cos \frac{\pi(2n+1)k}{2N} \qquad 1 \le k \le N-1 \quad (1)$$

Using A as a template, the ICT matrix $C = [c_{kn}]$ is an orthogonal matrix (i.e., $CC^t = \Delta$, where Δ is a diagonal matrix) with the following properties:

(1) Integer property: c_{kn} represents integers for $0 \le k$, $n \le N-1$.

(2) Orthogonality property: Rows (or columns) of C are orthogonal.

(3) Relationship with DCT:

(a) $sgn(c_{kn}) = sgn(a_{kn})$ for $0 \leq k,\ n \leq N-1$.

(b) If $a_{kn} = a_{st}$, then $c_{kn} = c_{st}$ for $0 \leq k, n, s, t \leq N-1$.

The integer property eliminates real multiplication and real addition operations. The orthogonality property assures that the inverse ICT has the same transform structure as the ICT. Notice that C is only required to be orthogonal, but not orthonormal. However, any orthogonal matrix can be made orthonormal by multiplying it by an appropriate diagonal matrix. This operation can be incorporated in the quantization (dequantization) stage of the compression (decompression) scheme, thus sparing the ICT (inverse ICT) from floating-point operations, and at the same time preserving the same transform structure as in the floating-point DCT (inverse DCT). The relationship between ICT and DCT guarantees efficient energy packing and allows the use of any fast DCT technique for the ICT.

V. ICT for $N = 8$

The floating-point 8×8 DCT matrix is shown in Fig. 2(a). A general structure of the 8×8 ICT matrix is given in Fig. 2(b). The symbols a, b, c, d, e, and f in Fig. 2(b) are numbers that satisfy conditions (1) through (3) given in Section IV. It was suggested in [6] to use $a = 5$, $b = 3$, $c = 2$, $d = 1$, $e = 3$, and $f = 1$ for the $N = 8$ ICT, as shown in Fig. 2(c). There are many other sets of (a, b, c, d, e, f) that can generate an orthogonal ICT. The integer set (a, b, c, d, e, f) gives an orthogonal ICT if, and only if, CC^t is a diagonal matrix. This is equivalent to the requirement

$$ab - ac - bd - cd = 0 \qquad (2)$$

with e and f arbitrary. The integer set (4,2,2,0,4,2) satisfies Eq. (2) and the corresponding ICT matrix is given in Fig. 2(d). Notice that the integers chosen are all powers of 2, and thus only simple binary additions and shift operations are required for this ICT. The MSE versus entropy performance of the compression scheme using this multiplication-free ICT is shown in Fig. 3. In view of the particular choice of the integers in the multiplication-free implementation, one expects the performance of this ICT to be inferior to that of the floating-point DCT and the ICT of [6]. However, the difference in the performance is small, as shown in Fig. 3.

VI. A General Procedure for Constructing an ICT Matrix

A general procedure to construct an $N \times N$ ICT matrix is presented in this section. For any $N \times N$ ICT matrix, this construction is done on the ground prior to implementation. The DCT matrix is used as a template to generate an ICT matrix. The procedure is as follows:

(1) Generate the $N \times N$ DCT matrix A.

(2) Construct an $N \times N$ matrix B by substituting the N possible absolute values in A with N symbols, and preserve the signs of the elements in A.

(3) Evaluate BB^t, and generate a set of independent algebraic equations that force BB^t to be a diagonal matrix.

(4) Find a set of N numbers that satisfy the set of algebraic equations generated in (3).

Since for a given N, there are $N(N-1)$ nondiagonal elements in C, part (3) of the procedure gives $N(N-1)/2$ quadratic equations. This set of equations is too large to be handled easily except for small N. However, by setting the most frequently occurring symbol in C to be an integer such as 1 or 2, the number of independent equations decreases substantially. As shown above, when $N = 8$, the number of equations is reduced from 28 to 1. The most tedious part of the above procedure is part (4), that is, finding N integers that satisfy the set of nonlinear algebraic equations generated in part (3). By using such advanced symbolic manipulation tools as *Mathematica* [7], the effort of generating the set of algebraic equations in part (3) and solving them in part (4) can be greatly reduced. In fact, *Mathematica* was used in an interactive manner to generate the 8×8 and 16×16 ICT matrices given in this article.

In order to obtain good compression performance, the set of $N-1$ integers must have a magnitude profile similar to the $N-1$ floating-point elements of A. Furthermore, if the multiplication-free property is desired, the set of N integers must be restricted to powers of 2. Some ad hoc techniques are usually needed to simplify the above calculations.

Note also that there is a general procedure for approximating an orthonormal matrix arbitrarily closely to one with rational coefficients. Given an orthonormal matrix Q with no eigenvalue equal to -1, let $S = (I-Q)(I+Q)^{-1}$. Then S is skew symmetric since

$$S + S^t = (I-Q)(I+Q)^{-1} + (I+Q^t)^{-1}(I-Q^t)$$

$$= (I - Q)(I + Q)^{-1} + (I + Q^{-1})^{-1}(I - Q^{-1})$$

$$= (I - Q)(I + Q)^{-1} + (I + Q)^{-1}(Q - I)$$

$$= 0 \tag{3}$$

Conversely, if S is any skew-symmetric matrix such that -1 is not an eigenvalue, then by essentially the same computation, the matrix $Q = (I - S)(I + S)^{-1}$ is orthonormal. Thus, given an orthonormal matrix Q, one can approximate $S = (I - Q)(I + Q)^{-1}$ by an arbitrarily close rational skew-symmetric matrix S'. Then, $Q' = (I - S')(I + S')^{-1}$ is a rational orthonormal matrix close to Q. While this procedure works well in theory, there are practical difficulties in its application. In practice, one considers $\lambda Q'$ where λ is an integer so that $\lambda Q'$ is integral. The matrix $\lambda Q'$ obtained by this procedure has generally large entries, which makes it unsuitable for many applications.

VII. ICT for $N = 16$

In this section, the general procedure of Section VI is used to construct a 16×16 ICT matrix. From Eq. (1), one obtains the 16×16 DCT matrix A. Notice that there are 16 non-negative values in A. The 16×16 ICT matrix B shown in Fig. 4(a) is obtained by using A as a template. By setting $a = 1$ and forcing all nondiagonal elements in BB^t to be zero, one obtains the following set of four independent nonlinear equations:

$$bc + cf - df - bg - eg - eh + di - hi = 0 \tag{4}$$

$$bd - ce - de + fg + bh - fh + ci + gi = 0 \tag{5}$$

$$-cd + be + bf - cg - dh + gh + ei - fi = 0 \tag{6}$$

$$jk - jl - km - lm = 0 \tag{7}$$

Notice that n and o are arbitrary. By extensive search on the sets of numbers that satisfy Eqs. (4), (5), (6), and (7), the following solution, which has a magnitude profile similar to that of the $N - 1$ floating-point elements of the 16×16 DCT matrix, was obtained: $a = 1$, $b = 18$, $c = 18$, $d = 16$, $e = 14$, $f = 14$, $g = 7$, $h = 10$, $i = 2$, $j = 10$, $k = 9$, $l = 6$, $m = 2$, $n = 56$, and $o = 2$. The corresponding ICT matrix is shown in Fig. 4(b). Another solution to the above system of equations is $a = 1$, $b = 4$, $c = 4$, $d = 0$, $e = 2$, $f = 2$, $g = 4$, $h = 0$, $i = 0$, $j = 4$, $k = 2$, $l = 2$, $m = 0$, $n = 1$, and $o = 4$. This matrix is shown in Fig. 4(c). Notice that all integers in this solution are powers of 2, so only binary shift and addition operations are required for this ICT transform. Since there are many zeros in this solution, one does not expect it to give an efficient ICT matrix. Intuitively, a transform matrix with good energy compaction should not have many zeros.

VIII. Conclusion

This article explored the mathematical properties of a new class of integer transforms called the integer cosine transform and derived a general construction procedure for this transform. This procedure can be used to construct integer versions of other transforms, such as the Fourier [8], sine [9], Gabor [10], wavelet [11], and so forth. The basic idea is to approximate a floating-point transform with its integer counterpart in the hope of achieving comparable performance with much-reduced implementation complexity. In the case of the discrete cosine transform, its integer counterpart, the ICT, has an implementation complexity substantially lower than that of the DCT and comparable to that of the Hadamard transform. Simulation results indicate that rate-distortion performance of the ICT is only slightly inferior to that of the DCT.

Acknowledgment

The authors thank I. Onyszchuk of the Jet Propulsion Laboratory for his helpful discussion.

References

[1] S. J. Dolinar, K.-M. Cheung, I. Onyszchuk, F. Pollara, and S. Arnold, "Compressed/Reconstructed Test Images for CRAF/Cassini," *TDA Progress Report 42-104*, vol. October–December 1990, Jet Propulsion Laboratory, Pasadena, California, pp. 88–97, February 15, 1991.

[2] *Joint Photographic Experts Group (JPEG) Draft Technical Specification (Rev. 5)*, International Standards Organization/International Consultative Committee for Telephone and Telegraph (ISO/CCITT), Washington, D.C., January 15, 1990.

[3] K.-M. Cheung, P. Smyth, and H. Wang, "A High-Speed Distortionless Predictive Image-Compression Scheme," *TDA Progress Report 42-102*, vol. April–June 1990, Jet Propulsion Laboratory, Pasadena, California, pp. 73–90, August 15, 1990.

[4] N. Ahmed, T. Natarajan, and K. Rao, "Discrete Cosine Transform," *IEEE Trans. Comput.*, vol. C-23, no. 1, pp. 90–93, January 1974.

[5] M. Narasimha and A. Peterson, "On the Computation of the Discrete Cosine Transform," *IEEE Trans. Comm.*, vol. COM-26, no. 6, pp. 934–946, June 1978.

[6] C. Choy, W. Cham, and L. Lee, "An Integer Cosine Transform Chip Using ASIC Technology," *Proceedings of the Picture Coding Symposium*, Torino, Italy, pp. 5.5.1–5.5.2, September 1988.

[7] S. Wolfram, *Mathematica: A System for Doing Mathematics by Computer*, Redwood City, California: Addison-Wesley Publishing Company, 1988.

[8] J. Cooley and J. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," *Math. of Comput.*, vol. 19, no. 90, pp. 297–301, April 1965.

[9] P. Yip and K. Rao, "A Fast Computational Algorithm for the Discrete Sine Transform," *IEEE Trans. Comm.*, vol. COM-28, no. 2, pp. 304–307, February 1980.

[10] D. Gabor, "Theory of Communications," *Institution of Electrical Engineering Journal*, vol. 93, no. 22, pp. 429–457, March 1946.

[11] A. Grossmann and J. Morlet, "Decomposition of Hardy Functions Into Square Integrable Wavelets of Constant Shape," *SIAM J. Math. Anal.*, vol. 15, no. 4, pp. 723–736, July 1984.

[12] F. Pollara and S. Arnold, "Emerging Standards for Still-Image Compression: A Software Implementation and Simulation Study," *TDA Progress Report 42-104*, vol. October–December 1990, Jet Propulsion Laboratory, Pasadena, California, pp. 98–102, February 15, 1991.

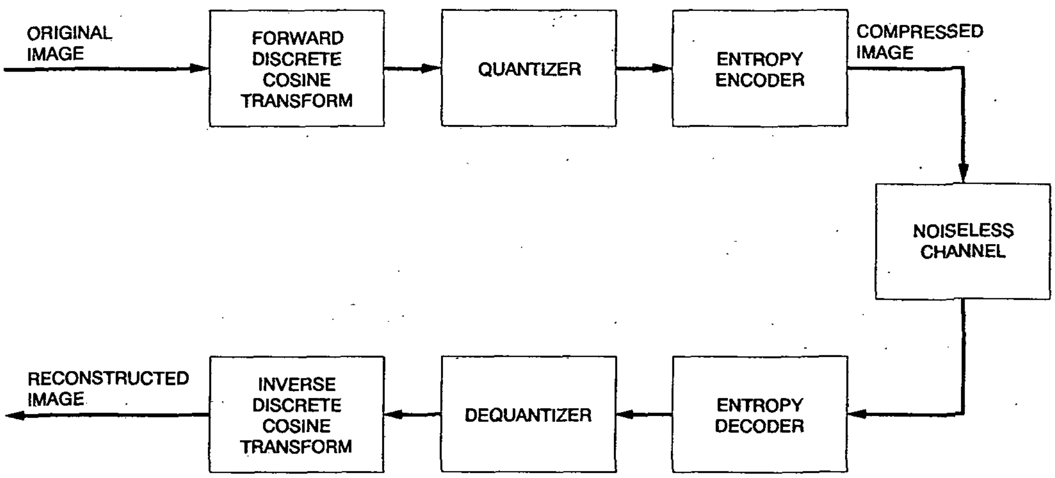

Fig. 1. DCT-based compression system.

(a)

$$A = \begin{matrix} 0.354 & 0.354 & 0.354 & 0.354 & 0.354 & 0.354 & 0.354 & 0.354 \\ 0.490 & 0.416 & 0.278 & 0.098 & -0.098 & -0.278 & -0.416 & -0.490 \\ 0.462 & 0.191 & -0.191 & -0.462 & -0.462 & -0.191 & 0.191 & 0.462 \\ 0.416 & -0.098 & -0.490 & -0.278 & 0.278 & 0.490 & 0.098 & -0.416 \\ 0.354 & -0.354 & -0.354 & 0.354 & 0.354 & -0.354 & -0.354 & 0.354 \\ 0.278 & -0.490 & 0.098 & 0.416 & -0.416 & -0.098 & 0.490 & -0.278 \\ 0.191 & -0.462 & 0.462 & -0.191 & -0.191 & 0.462 & -0.462 & 0.191 \\ 0.098 & -0.278 & 0.416 & -0.490 & 0.490 & -0.416 & 0.278 & -0.098 \end{matrix}$$

(b)

$$B = \begin{matrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ a & b & c & d & -d & -c & -b & -a \\ e & f & -f & -e & -e & -f & f & e \\ b & -d & -a & -c & c & a & d & -b \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ c & -a & d & b & -b & -d & a & -c \\ f & -e & e & -f & -f & e & -e & f \\ d & -c & b & -a & a & -b & c & -d \end{matrix}$$

(c)

$$C = \begin{matrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 5 & 3 & 2 & 1 & -1 & -2 & -3 & -5 \\ 3 & 1 & -1 & -3 & -3 & -1 & 1 & 3 \\ 3 & -1 & -5 & -2 & 2 & 5 & 1 & -3 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 2 & -5 & 1 & 3 & -3 & -1 & 5 & -2 \\ 1 & -3 & 3 & -1 & -1 & 3 & -3 & 1 \\ 1 & -2 & 3 & -5 & 5 & -3 & 2 & -1 \end{matrix}$$

(d)

$$D = \begin{matrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 4 & 2 & 2 & 0 & 0 & -2 & -2 & -4 \\ 4 & 2 & -2 & -4 & -4 & -2 & 2 & 4 \\ 2 & 0 & -4 & -2 & 2 & 4 & 0 & -2 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 2 & -4 & 0 & 2 & -2 & 0 & 4 & -2 \\ 2 & -4 & 4 & -2 & -2 & 4 & -4 & 2 \\ 0 & -2 & 2 & -4 & 4 & -2 & 2 & 0 \end{matrix}$$

Fig. 2. Four 8×8 transform matrices: (a) DCT matrix (real number entities rounded off to nearest thousandth); (b) structure of ICT matrix; (c) ICT matrix from [6]; and (d) multiplication-free ICT.

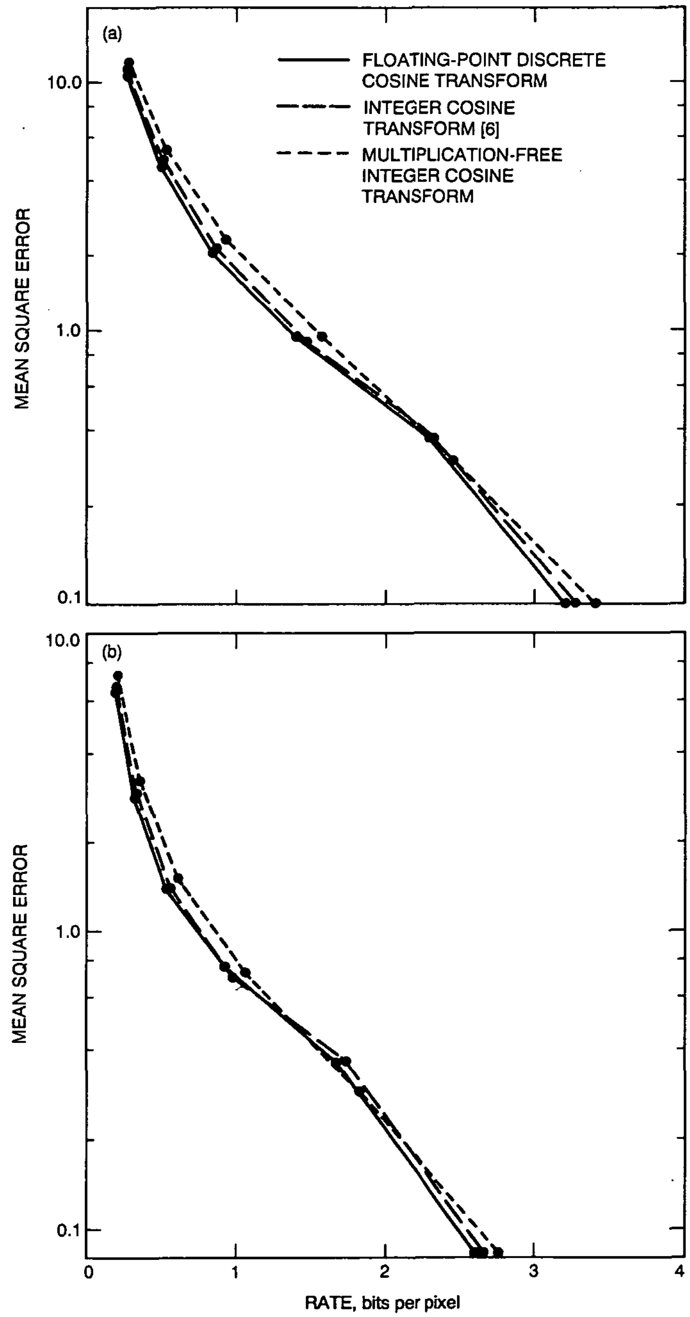

Fig. 3. Rate-distortion performance of three transform algorithms: (a) image *saturn1*, and (b) image *saturn2*.

(a)

a	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a
b	c	d	e	f	g	h	i	-i	-h	-g	-f	-e	-d	-c	-b
j	k	l	m	-m	-l	-k	-j	-j	-k	-l	-m	m	l	k	j
c	f	i	-g	-d	-b	-e	-h	h	e	b	d	g	-i	-f	-c
o	n	-n	-o	-o	-n	n	o	o	n	-n	-o	-o	-n	n	o
d	i	-e	-c	-h	f	b	g	-g	-b	-f	h	c	e	-i	-d
k	-m	-j	-l	l	j	m	-k	-k	m	j	l	-l	-j	-m	k
e	-g	-c	i	b	h	-d	-f	f	d	-h	-b	-i	c	g	-e
a	-a	-a	a	a	-a	-a	a	a	-a	-a	a	a	-a	-a	a
f	-d	-h	b	-i	-c	g	e	-e	-g	c	i	-b	h	d	-f
l	-j	m	k	-k	-m	j	-l	-l	j	-m	-k	k	m	-j	l
g	-b	f	h	-c	e	i	-d	d	-i	-e	c	-h	-f	b	-g
n	-o	o	-n	-n	o	-o	n	n	-o	o	-n	-n	o	-o	n
h	-e	b	-d	g	i	-f	c	-c	f	-i	-g	d	-b	e	-h
m	-l	k	-j	j	-k	l	-m	-m	l	-k	j	-j	k	-l	m
i	-h	g	-f	e	-d	c	-b	b	-c	d	-e	f	-g	h	-i

(b)

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
18	18	16	14	14	7	10	2	-2	-10	-7	-14	-14	-16	-18	-18
10	9	6	2	-2	-6	-9	-10	-10	-9	-6	-2	2	6	9	10
18	14	2	-7	-16	-18	-14	-10	10	14	18	16	7	-2	-14	-18
2	5	-5	-2	-2	-5	5	2	2	5	-5	-2	-2	-5	5	2
16	2	-14	-18	-10	14	18	7	-7	-18	-14	10	18	14	-2	-16
9	-2	-10	-6	6	10	2	-9	-9	2	10	6	-6	-10	-2	9
14	-7	-18	2	18	10	-16	-14	14	16	-10	-18	-2	18	7	-14
1	-1	-1	1	1	-1	-1	1	1	-1	-1	1	1	-1	-1	1
14	-16	-10	18	-2	-18	7	14	-14	-7	18	2	-18	10	16	-14
6	-10	2	9	-9	-2	10	-6	-6	10	-2	-9	9	2	-10	6
7	-18	14	10	-18	14	2	-16	16	-2	-14	18	-10	-14	18	-7
5	-2	2	-5	-5	2	-2	5	5	-2	2	-5	-5	2	-2	5
10	-14	18	-16	7	2	-14	18	-18	14	-2	-7	16	-18	14	-10
2	-6	9	-10	10	-9	6	-2	-2	6	-9	10	-10	9	-6	2
2	-10	7	-14	14	-16	18	-18	18	-18	16	-14	14	-7	10	-2

(c)

2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
4	4	0	2	2	4	0	0	0	0	-4	-2	-2	0	-4	-4
4	2	2	0	0	-2	-2	-4	-4	-2	-2	0	0	2	2	4
4	2	0	-4	0	-4	-2	0	0	2	4	0	4	0	-2	-4
4	1	-1	-4	-4	-1	1	4	4	1	-1	-4	-4	-1	1	4
0	0	-2	-4	0	2	4	4	-4	-4	-2	0	4	2	0	0
2	0	-4	-2	2	4	0	-2	-2	0	4	2	-2	-4	0	2
2	-4	-4	0	4	0	0	-2	2	0	0	-4	0	4	4	-2
2	-2	-2	2	2	-2	-2	2	2	-2	-2	2	2	-2	-2	2
2	0	0	4	0	-4	4	2	-2	-4	4	0	-4	0	0	-2
2	-4	0	2	-2	0	4	-2	-2	4	0	-2	2	0	-4	2
4	-4	2	0	-4	2	0	0	0	0	-2	4	0	-2	4	-4
1	-4	4	-1	-1	4	-4	1	1	-4	4	-1	-1	4	-4	1
0	-2	4	0	4	0	-2	4	-4	2	0	-4	0	-4	2	0
0	-2	2	-4	4	-2	2	0	0	2	-2	4	-4	2	-2	0
0	0	4	-2	2	0	4	-4	4	-4	0	-2	2	-4	0	0

Fig. 4. Three 16 × 16 ICT matrices: (a) structure of ICT matrix; (b) ICT matrix; and (c) multiplication-free ICT matrix.