Development of integer cosine transforms by the principle of dyadic symmetry

W.-K. Cham, PhD

Indexing terms: Image processing, Codes and decoding

Abstract: The paper shows how to convert the order-8 cosine transforms into a family of integer cosine transforms (ICTs) using the theory of dyadic symmetry. The new transforms can be implemented using simple integer arithmetic. It was found that performance close to that of the DCT can be achieved with an ICT that requires only 4 bits for representation of its kernel component magnitude. Better performance can be achieved by some ICTs whose kernel components require longer bit lengths for representation. ICTs that require 3 bits or less for representation of their component magnitude are available but with degraded performance. The availability of many ICTs provides an engineer the freedom to tradeoff performance for simple implementation in designing a transform codec.

1 Introduction

Transform coding can achieve a high data compression rate for image data. A transform coder comprises mainly two parts: the first part transforms highly correlated image data into weakly correlated coefficients using an orthogonal transform and the second part performs adaptive quantisation on coefficients to reduce the bit transmission rate. It has been widely accepted that among the many suboptimal orthogonal transforms the discrete cosine transform (DCT) has the best performance in both data compression and filtering for image data [1, 2]. The DCT has also been shown to be asymptotically optimal for a first-order Markov source [3, 4], which is regarded as a good stochastic representation of image data. As block sizes of 8 and 16 are most appropriate for the transform coding of image data, therefore techniques for implementing order-8 and -16 DCTs economically and with fast computational time are very important for the realisation of a transform codec.

The DCT can be implemented using either a programmable processor or dedicated hardware. The availability of dedicated hardware that computes the DCT using floating point arithmetic is very complex and expensive. On the other hand, a programmable processor which computes the DCT by executing a sequence of instructions does not usually have instructions that handle real numbers. Although floating point co-processors which have been specially tailored for processing floating point arithmetic can be found on the commercial market, they are also very expensive.

A simple method to eliminate floating point arithmetic is to approximate the real magnitudes of the DCT components by M-bit integers, so that the DCT can be computed using integer multiplications and additions. With each pixel represented by 8 bits, the 1-D order-8 DCT requires $n^2 (8 + M)$-bit multiplication and $(8 + M + \frac{1}{2} \log_2 n)$-bit addition operations. The second stage of the 2-D DCT requires $n^2 (8 + 2M + \frac{1}{2} \log_2 n)$-bit multiplication and $(8 + 2M + \log_2 n)$-bit addition operations.

Guglielmo has suggested that 7 bits should be enough to represent the magnitudes of order-16 DCT kernel components without causing significant effects on the transformed and reconstructed signal [5]. Suppose the order-8 DCT also requires 7 bits for representation of the components. The 2-D order-8 DCT thus requires 24-bit multiplication and 25-bit addition operations, which are difcult to implement and introduce a lot of delay in the computation. There are other, simpler transforms, such as the Walsh transform [6, 7], slant transform [8] and the high-correlation transform (HCT) [9], whose multiplication and addition operations require shorter bit lengths, but they all perform unsatisfactorily compared to the DCT.

Jones et al. [10] found that the order-8 DCT can be approximated using the orthogonal C-matrix transform (CMT) with small performance degradation. The C-matrix transform is computed via the row bit-reversed Walsh transform $[H]$ and the C-matrix $[CM]$ as follows:

$$[CM] = [CM] \cdot [H]$$

As $[CM]$ is a sparse block diagonal matrix containing only integers 13, 12, 5, 4, 3, -3, -4 and -5, and as $[H]$ contains only +1 and -1, the C-matrix transform can be implemented using simple integer arithmetic. The work was then extended to orders 16 [11] and 32 [12]. These three C-matrices are all derived by trial and error.

In this paper, we show that it is possible to replace transform kernel components of the order-8 DCT by a new set of numbers. Such a technique can be applied to any transform, although solutions are not always guaranteed. For the order-8 DCT, infinite new transforms can be obtained. Boundary conditions are imposed to ensure
that the new transforms, while still resembling the DCT, remain}{

that the new transforms, while still resembling the DCT, remain

DCT, and the magnitude of these integers can be very small, the new transforms thus generated are

simple to implement and will be referred to here as

The sign of each component of the ICTs is the same as

that of the DCT whereas the sign of each component of

the ICTs is the same as that of the Walsh transform. Therefore, the basis vectors of the ICTs are closer to

those of the DCT than the CMT. To access the performance of the ICTs, the transform efficiency [9] of a

2-D Markov source and real images are used as criteria.

It was found that the performance of most of the ICTs was very close to that of the DCT and better than that of the

CMT. To access the performance of the ICTs, the transform efficiency [9] of a

2-D Markov source and real images are used as criteria.

The availability of many ICTs therefore provides an engi-

neer with the freedom to tradeoff performance for simple

implementation in the design of a transform codec. For

example, if implementation simplicity is the paramount

criterion, he can choose an ICT whose components can

be represented using only 2 bits.

2 Dyadic symmetry

Definition of dyadic symmetry:

A vector of $2^n$ elements $[a_0, a_1, \ldots, a_{2^n-1}]$ is said to

have the $i$th dyadic symmetry if and only if $a_j = s \cdot a_{j+1}$,

where $\oplus$ is the 'exclusive or' operation, $j$ lies in the range

$[0, 2^n - 1]$ and $i$ in the range $[1, 2^n - 1]$. $s = 1$ when the

symmetry is even, and $s = -1$ when the symmetry is

odd.

For a vector of eight elements, there are seven possible
dyadic symmetries. As an example, Table 1 shows the

Table 1: The seven vectors $H_i$ having $i$th even dyadic symmetry

$$
S \quad \text{Vector } H_i \\
\begin{array}{|c|c|c|c|c|c|c|c|}
\hline
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
3 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
4 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
5 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
6 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
7 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\hline
\end{array}
$$

vectors $H_i = [a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7]$ which have the

seven even dyadic symmetries.

Theorem of orthogonality: Two vectors $U$ and $V$ are

orthogonal if $U$ and $V$ have the same type of dyadic symmetry and one is even and the other is odd.

Proof: It is obvious that the dot product of vectors $U$

and $V$ are zero, therefore $U$ and $V$ are orthogonal.

3 Generation of the order-$8$ ICTs

Let $[T]$ be the kernel of the order-$n$ discrete cosine transform whose $(i, j)$th element is $T_{ij}(i, j)$. $T_{ij}(i, j)$ is the $j$th component of the $i$th DCT basis vector and is equal to

$$
T_{ij}(i, j) = \frac{1}{\sqrt{n}} \cos \left( \frac{(i + 0.5)n}{n} \right) \quad \text{for } i = 0 \quad \text{and } j \in [0, n]
$$

where

$$
j \in [0, n - 1]
$$

The following describes the steps to convert the order-$8$

DCT kernel into ICT kernels.

Step 1 to express the order-$8$ DCT kernel

in the form of a matrix of variables: Let $n$ of eqn. 1 be 8 so that we obtain an $8 \times 8$ matrix

$$
[T] = \begin{bmatrix}
k_0 & k_1 & k_2 & k_3 & k_4 & k_5 & k_6 & k_7 \\
k_4 & k_5 & k_6 & k_7 & k_0 & k_1 & k_2 & k_3 \\
k_8 & k_9 & k_{10} & k_{11} & k_0 & k_1 & k_2 & k_3 \\
\end{bmatrix}
$$

where $J_i$ is the $i$th basis vector and $k_i$ is a scaling constant such that $|J_i| = 1$. Let $J_i$ be the $i$th element of $J_i$. For simplicity, we shall denote $T_{i,(i)}$ by $T_{i, j}$. As $|T(1, 0)| = |J(1, 7)| = |J(3, 2)| = |J(5, 7)|$, we may represent the magnitudes of $|J(1, 0)|, |J(1, 7)|, |J(3, 2)|, |J(5, 7)|, |J(5, 6)|, |K(7, 3)|$ and $|J(7, 4)|$ by a single variable, say $'a'$. Similarly, all eight basis vectors are expressed as variables '$a', 'b',

'c', 'd', 'e' and 'f' as shown in Table 2. Suppose $k_i$'s are

chosen such that $d$ and $f$ are unity, then $a, b, c$ and $e$

are variables representing irrational numbers whose values

are approximately 5.027, 4.2620, 2.8478 and 2.4142

respectively.

Step 2 to find the conditions under which $J_i$ and $J_j$

are orthogonal: From Table 2, we can see that each basis

vector has at least one dyadic symmetry. Table 3 lists the

type of dyadic symmetry present in each basis vector. We

then examine the condition under which the $i$th basis

vector $J_i$ and the $j$th basis vector $J_j$ are orthogonal for

all $i, j$ and $i/j$. For example, we can see that $J_0$ and $J_1$

are always orthogonal to each other for all $a, b, c$ and $d$

because $J_0$ has odd 7th dyadic symmetry and $J_0$ has even

7th dyadic symmetry. Therefore, by means of the theorem

of orthogonality, $J_0$ and $J_1$ are always orthogonal. $J_0$

and $J_2$ are also always orthogonal for all $e$ and $f$ because
Table 3: $S$th dyadic symmetry type in basis vector $J_i$

<table>
<thead>
<tr>
<th>$i$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>O</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>-</td>
<td>O</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>E</td>
</tr>
<tr>
<td>3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>O</td>
</tr>
<tr>
<td>4</td>
<td>O</td>
<td>O</td>
<td>E</td>
<td>E</td>
<td>O</td>
<td>O</td>
<td>E</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>O</td>
</tr>
<tr>
<td>6</td>
<td>-</td>
<td>-</td>
<td>O</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>E</td>
</tr>
<tr>
<td>7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>O</td>
</tr>
</tbody>
</table>

$E$ and $O$ represent even and odd dyadic symmetry respectively. '-' implies that $J_i$ has no $S$th dyadic symmetry.

$J_2$ has odd 3rd dyadic symmetry and $J_0$ has even 3rd dyadic symmetry. In another example, we can see that $J_1$ and $J_2$ are orthogonal to each other for all $a$, $b$, $c$, $d$, $e$ and $f$ because $J_1$ has odd 7th dyadic symmetry while $J_2$ has even 7th dyadic symmetry. The conditions under which $J_i$ and $J_j$ are orthogonal are summarised in Table 4. Table 4 reveals that the only condition that the constants $a$, $b$, $c$, $d$ and $e$ must satisfy to ensure that the transform $[T]$ be orthogonal is

$$a \cdot b = a \cdot c + b \cdot d + c \cdot d$$  \hspace{1cm} (3)

Eqn. 3 has four variables and has an infinite number of solutions. This implies that an infinite number of new orthogonal transforms can be generated from the DCT.

Step 3 to set up boundary conditions and generate new transforms: Eqn. 1 implies that for the DCT

$$a \geq b \geq c \geq d \quad \text{and} \quad e \geq f \quad \text{(4)}$$

To make the basis vectors of the new transforms resemble those of the DCT, inequality exprs. 4 have to be satisfied. Furthermore, to eliminate truncation error due to non-exact representation of the basis components $a$, $b$, $c$, $d$, $e$ and $f$, condition expr. 5 has to be satisfied:

$$a, b, c, d, e \text{ and } f \text{ are integers} \quad \text{(5)}$$

Transforms $[T]$ that satisfy the conditions of exprs. 3, 4 and 5 are referred to here as order-8 integer cosine transforms (ICTs). As an example, ICT(5, 3, 2, 1, 3, 1) refers to the order-8 ICT with $a = 5$, $b = 3$, $c = 2$, $d = 1$, $e = 3$, $f = 1$. Fig. 1 shows the basis vectors of this ICT together with those of the DCT.

4 Performance of order-8 ICTs

4.1 Transform efficiency [9] performance

In the transform coding of pictures, transforms are used to convert highly correlated signals into coefficients of low correlation. Such decorrelation ability may be measured by the transform efficiency $\eta$, which is defined on a first-order Markov process of adjacent element correlation $\rho$. A large $\eta$ implies a high decorrelation ability. The optimal K-L transform which converts signals into completely uncorrelated coefficients has a transform efficiency equal to 100% for all $\rho$.

Let the $n$-dimensional vector $X$ be a sample from a one-dimensional, zero-mean, unit-variance first-order Markov process with adjacent element correlation $\rho$, and covariance matrix $[C_X]$ where the $(i, j)$th element of $[C_X]$ is $\rho^{i-j}$. The efficiency of the transform $[T]$ is defined on the transform domain covariance matrix $[C_Y]$ of vector $Y$ where

$$Y = [T]X$$

$$[C_Y] = [T][C_X][T]^T$$

$$= \begin{bmatrix} s_{11} & \cdots & s_{1n} \\ \vdots & \ddots & \vdots \\ s_{n1} & \cdots & s_{nn} \end{bmatrix}$$

Efficiency $\eta$ is defined as

$$\eta = \frac{\sum_{i=1}^{n} |s_{ii}|}{\sum_{p=1}^{n} \sum_{q=1}^{n} |s_{pq}|}$$

The DCT, which is widely accepted as the best suboptimal transform, has the highest transform efficiency of the well-known suboptimal transforms for $\rho$ close to unity. A computer search has been performed to find the set of $(a, b, c, d)$ that gives the highest transform efficiency for $a$ less than or equal to 255, and $(e, f)$ equal to $(1, 0), (4, 1), (3, 1), (2, 1)$ and $(1, 1)$. It was found that $e = 3$ and $f = 1$ always gives a higher transform efficiency for the same $(a, b, c, d)$. Table 5 lists the twelve order-8 ICTs that
have the highest transform efficiencies for $p$ equal to 0.9 and $a$ less than or equal to 255. It can be seen that all twelve ICTs have high transform efficiencies than the

<table>
<thead>
<tr>
<th>Transform</th>
<th>Transform efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICT(230, 201, 134, 46, 3, 1)</td>
<td>90.221</td>
</tr>
<tr>
<td>ICT(175, 153, 102, 35, 3, 1)</td>
<td>90.220</td>
</tr>
<tr>
<td>ICT(120, 105, 70, 24, 3, 1)</td>
<td>90.219</td>
</tr>
<tr>
<td>ICT(185, 162, 108, 37, 3, 1)</td>
<td>90.217</td>
</tr>
<tr>
<td>ICT(250, 219, 146, 50, 3, 1)</td>
<td>90.217</td>
</tr>
<tr>
<td>ICT(65, 57, 38, 13, 3, 1)</td>
<td>90.217</td>
</tr>
<tr>
<td>ICT(10, 9, 6, 2)</td>
<td>90.217</td>
</tr>
<tr>
<td>ICT(55, 48, 32, 11, 3, 1)</td>
<td>90.217</td>
</tr>
<tr>
<td>ICT(205, 180, 120, 41, 3, 1)</td>
<td>90.217</td>
</tr>
<tr>
<td>ICT(140, 123, 82, 28, 3, 1)</td>
<td>90.217</td>
</tr>
<tr>
<td>ICT(215, 189, 126, 43, 3, 1)</td>
<td>90.217</td>
</tr>
<tr>
<td>ICT(75, 66, 44, 15, 3, 1)</td>
<td>90.217</td>
</tr>
<tr>
<td>ICT(235, 207, 138, 47, 3, 1)</td>
<td>90.217</td>
</tr>
</tbody>
</table>

$B$ Basis restriction mean-square-error performance

The data compression ability of a transform can be measured by means of the basis restriction mean-square-error.

![Fig. 2](transform_efficiency.png)

**Fig. 2** Transform efficiency for different values of adjacent element correlation

- △ DCT
- × Walsh transform
- ○ ICT(230, 201, 134, 46)
- ◇ ICT(153, 48, 32, 11)
- □ ICT(65, 57, 38, 13)
- ▽ CMT

Transform order-8 DCT. Fig. 2 shows how the transform efficiency of various transforms varies with adjacent element correlation $p$. As shown in Fig. 2, the transform efficiencies of the DCT and the ICTs are very close to each other and are always better than those of the CMT and the Walsh transform for adjacent element coefficients between 0.1 and 0.9.

![Fig. 3](basis_restriction.png)

**Fig. 3** Basis restriction mean-square-errors for adjacent element correlation $p$ equal to 0.95

- □ ICTs
- ◇ KLT
- ◆ DCT
- × Walsh transform
- ▽ CMT

Consider a two-dimensional zero-mean unit-variance nonseparable isotropic Markov process with covariance function

$$C_s(i, j; p, q) = E[x_{i,j} \cdot x_{p, q}] = \rho e^{-(i-p)^2+(j-q)^2}$$

where $\rho$ is the adjacent element correlation in the vertical and horizontal directions. Let the $n \times n$ matrix $[X]$ be a sample of the Markov process. Suppose $[X]$ is transformed into $[C]$ by transform $[T]$, i.e.

$$[C] = [T] \cdot [X] \cdot [T]^\dagger$$

where the elements of $[X]$ and $[C]$ are $x_{i,j}$ and $c_{u,v}$, respectively. The covariance function of $[C]$ is

$$c_{u,v} = E[x_{u,v} \cdot c_{r,s}] = \sum_{(i, j) \in R} C_s(i, j; p, q)c_{r,s}$$

where $R$ is the set containing $M$ index pairs $(u, v)$ corresponding to the largest $M$ $c_{u,v}$. Hence, we have the variance of $c_{u,v}$ equal to

$$\sigma^2(u, v) = c_{u,v}^2$$

Let $\Omega$ be the set containing $M$ index pairs $(u, v)$ corresponding to the largest $M$ $\sigma^2(u, v)$. The basis restriction

<table>
<thead>
<tr>
<th>no. of coefficients retained</th>
<th>KLT</th>
<th>DCT</th>
<th>ICT(230, 201, 134, 46, 3, 1)</th>
<th>ICT(55, 48, 32, 11, 3, 1)</th>
<th>ICT(10, 9, 6, 2, 1)</th>
<th>CMT</th>
<th>Walsh</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.1372</td>
<td>0.1381</td>
<td>0.1381</td>
<td>0.1381</td>
<td>0.1382</td>
<td>0.1387</td>
<td>0.1468</td>
</tr>
<tr>
<td>6</td>
<td>0.0567</td>
<td>0.0572</td>
<td>0.0573</td>
<td>0.0573</td>
<td>0.0573</td>
<td>0.0587</td>
<td>0.0785</td>
</tr>
<tr>
<td>10</td>
<td>0.0406</td>
<td>0.0410</td>
<td>0.0410</td>
<td>0.0410</td>
<td>0.0410</td>
<td>0.0431</td>
<td>0.0541</td>
</tr>
<tr>
<td>14</td>
<td>0.0320</td>
<td>0.0322</td>
<td>0.0323</td>
<td>0.0323</td>
<td>0.0323</td>
<td>0.0348</td>
<td>0.0441</td>
</tr>
<tr>
<td>18</td>
<td>0.0263</td>
<td>0.0264</td>
<td>0.0266</td>
<td>0.0266</td>
<td>0.0266</td>
<td>0.0287</td>
<td>0.0361</td>
</tr>
<tr>
<td>22</td>
<td>0.0221</td>
<td>0.0222</td>
<td>0.0223</td>
<td>0.0223</td>
<td>0.0224</td>
<td>0.0238</td>
<td>0.0300</td>
</tr>
<tr>
<td>26</td>
<td>0.0189</td>
<td>0.0190</td>
<td>0.0190</td>
<td>0.0190</td>
<td>0.0190</td>
<td>0.0198</td>
<td>0.0251</td>
</tr>
<tr>
<td>30</td>
<td>0.0160</td>
<td>0.0160</td>
<td>0.0162</td>
<td>0.0162</td>
<td>0.0162</td>
<td>0.0165</td>
<td>0.0235</td>
</tr>
<tr>
<td>34</td>
<td>0.0136</td>
<td>0.0136</td>
<td>0.0137</td>
<td>0.0137</td>
<td>0.0137</td>
<td>0.0140</td>
<td>0.0170</td>
</tr>
</tbody>
</table>
Table 7: Mean-square-errors (MSE) of the ICTs that have the highest transform efficiencies

<table>
<thead>
<tr>
<th>Transform or ICT(a, b, c, d)</th>
<th>Transform efficiency</th>
<th>MSE of 'Girl'</th>
<th>MSE of 'House'</th>
<th>MSE of 'Stone'</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCT</td>
<td>89.8</td>
<td>10.6</td>
<td>22.3</td>
<td>17.3</td>
</tr>
<tr>
<td>CMT</td>
<td>86.8</td>
<td>10.8</td>
<td>22.9</td>
<td>17.7</td>
</tr>
<tr>
<td>slant transform</td>
<td>85.8</td>
<td>10.6</td>
<td>22.7</td>
<td>17.7</td>
</tr>
<tr>
<td>HCT</td>
<td>84.1</td>
<td>12.0</td>
<td>24.8</td>
<td>19.2</td>
</tr>
<tr>
<td>Walsh transform</td>
<td>77.1</td>
<td>15.8</td>
<td>32.3</td>
<td>25.1</td>
</tr>
</tbody>
</table>

* for ICT(a, b, c, d) a is 255 or less and needs 8 bits or less for representation

| ICT(230, 201, 134, 46)      | 90.2                 | 10.8         | 22.4           | 17.3          |
| ICT(175, 102, 35)          | 90.2                 | 10.8         | 22.4           | 17.3          |
| ICT(120, 70, 24)           | 90.2                 | 10.8         | 22.4           | 17.3          |
| ICT(185, 162, 108, 37)     | 90.2                 | 10.8         | 22.4           | 17.3          |
| ICT(250, 219, 146, 30)     | 90.2                 | 10.8         | 22.4           | 17.3          |

* for ICT(230, 201, 134, 46), ICT(175, 102, 35) and ICT(120, 70, 24) a is 127 or less and needs 7 bits or less for representation

| ICT(55, 48, 32, 11)        | 90.2                 | 10.8         | 22.4           | 17.3          |
| ICT(120, 105, 70, 24)      | 90.2                 | 10.8         | 22.4           | 17.3          |
| ICT(185, 162, 108, 37)     | 90.2                 | 10.8         | 22.4           | 17.3          |
| ICT(250, 219, 146, 30)     | 90.2                 | 10.8         | 22.4           | 17.3          |

* for ICT(55, 48, 32, 11), ICT(120, 105, 70, 24), ICT(185, 162, 108, 37) and ICT(250, 219, 146, 30) a is 63 or less and needs 6 bits or less for representation

| ICT(10, 9, 6, 2)           | 90.2                 | 10.8         | 22.5           | 17.3          |
| ICT(55, 31, 14, 7)         | 90.1                 | 10.9         | 22.6           | 17.4          |
| ICT(45, 39, 26, 9)         | 90.1                 | 10.7         | 22.4           | 17.3          |
| ICT(45, 42, 28, 9)         | 90.1                 | 10.9         | 22.6           | 17.4          |

* for ICT(10, 9, 6, 2), ICT(55, 31, 14, 7) and ICT(45, 39, 26, 9) a is 31 or less and needs 5 bits or less for representation

| ICT(10, 9, 6, 2)           | 90.2                 | 10.8         | 22.5           | 17.3          |
| ICT(25, 24, 16, 6)         | 89.9                 | 11.0         | 22.7           | 17.4          |
| ICT(25, 21, 14, 5)         | 89.8                 | 10.7         | 22.4           | 17.3          |
| ICT(24, 21, 15, 4)         | 89.6                 | 10.8         | 22.5           | 17.3          |
| ICT(26, 24, 15, 6)         | 89.6                 | 10.8         | 22.6           | 17.4          |

* for ICT(10, 9, 6, 2), ICT(25, 24, 16, 6) and ICT(25, 21, 14, 5) a is 15 or less and needs 4 bits or less for representation

| ICT(6, 6, 3, 2)            | 83.2                 | 11.3         | 23.4           | 18.0          |
| ICT(5, 3, 2, 1)            | 81.1                 | 11.0         | 23.4           | 18.3          |
| ICT(7, 4, 3, 1)            | 80.0                 | 11.0         | 23.6           | 18.4          |
| ICT(5, 2, 1, 1)            | 80.0                 | 11.5         | 24.3           | 19.0          |

\[ e = 3, f = 1 \]

mean-square-error is defined as

\[ e(M) = 1 - \sum_{a,b,c,d} \sigma_{a,b,c,d} (u, v)^2 \sum_{a,b,c,d} \sigma_{a,b,c,d} (u, v)^2 \]

Fig. 3 and Table 6 show comparisons of the basis restriction mean-square-errors of various transforms for \( \rho \) equal to 0.95. The ICTs being tested are ICT(230, 201, 134, 46), ICT(55, 48, 32, 11, 3, 1) and ICT(10, 9, 6, 2, 3, 1) whose basis restriction mean-square-errors, as shown in Table 6, are nearly the same and so are represented using one curve in Fig. 3. Fig. 3 shows that the basis restriction mean-square-errors of the ICTs, the KLT and the DCT are very close and always smaller than those of the CMT and the Walsh transform. Table 6 shows that the basis restriction mean-square-error of the ICT is in fact smaller than that of the DCT which in turn is smaller than those of the ICTs.

### 4.3 Mean-square-error performance on real images

Tests have also been performed using real images. Images are first divided to form horizontal order-8 vectors and then transformed. With five high-sequence coefficients truncated, the transformed vectors are inverse transformed back into the spatial domain. The mean-square-error between the original and the processed image is used as the criterion for assessing the performance of a transform. Table 7 lists the results for the mean-square-error test performed on the three images shown in Fig. 4.

![Figure 4](image-url)  

**Fig. 4** The three images used in the mean-square-error test
- a Girl
- b House
- c Stone

It was found that the mean-square-error performance of the transforms varies slightly from image to image. For image 'Girl', the slant transform has the smallest mean-square-error and is followed by ICT(12, 10, 6, 3, 3, 1), the DCT and then ICT(15, 12, 8, 3, 3, 1). For image 'House', the DCT has the smallest mean-square-error and is followed by ICT(15, 12, 8, 3, 3, 1) and ICT(25, 21, 14, 5, 3, 1). For image 'Stone', ICT(25, 21, 14, 5, 3, 1), ICT(45, 39, 26, 9, 3, 1) and ICT(55, 48, 32, 11, 3, 1) produce the smallest mean-square-errors.

ICT(10, 9, 6, 2, 3, 1), whose implementation complexity is about the same as that of the CMT, has smaller mean-
square-errors than the CMT, for all three images. The Walsh transform, which has the smallest transform efficiency and the largest basis restriction mean-square-error, has the largest mean-square-error for all three images. In general, transforms having larger transform efficiencies and smaller basis restriction mean-square-errors usually have smaller mean-square-errors in the real-image test. However, it should be noted that results drawn from the real-image test are only used to confirm the results based on the stochastic processes and should not be generalised.

5 Implementation

Let \([T]\) be an ICT. From eqn. 2, we have

\[
[T]^{-1} = [T][K]
\]

where \([K]\) is a diagonal matrix whose \((i, i)\)th element is \(k_i\) and \([J]\) is a matrix whose \(i\)th basis vector is given by Table 2. As \([T]\) is an ICT, therefore \([J]\) contains only integers. Fig. 5 shows an adaptive 1-D transform coding system that utilises an order-8 ICT. In the transmitter, signal vector \(X\) is first transformed by \([J]\) and then each transformed coefficient is multiplied by the corresponding \(k_i\) to form \(c_i\). Coefficient \(c_i\) is then quantised by quantiser \(\# i\) under the control of an adaptive scheme to form the quantised coefficient \(c'_i\), which is then multiplexed to the channel. At the receiver, each \(c'_i\) is demultiplexed and decoded separately into a form suitable for the multiplication process and the inverse transform. The quantisation process at the transmitter and the decoding process at the receiver are most likely performed using microprocessors by means of table-lookup. This implies that the multiplication processes at the transmitter can be easily incorporated into the quantisation process by modifying the input entries of the quantisation lookup table; the multiplication processes at the receiver can be incorporated into the decoding process by modifying the output entries of the decoding lookup table. Therefore, real number multiplications may be completely eliminated in an ICT transform coder.

To speed up the transformation process, the integer transform \([J]\) should be implemented using a fast computational algorithm which is very similar to that of the DCT [3] and has 4 iterations.

6 Generalisation to larger block sizes

An order-\(n\) orthogonal transform \(T_{2n}(i, j)\) can be generated from an order-\(n\) \(T_d(i, j)\) transform as follows:

(a) the first \(n\) basis vectors of \(T_{2n}(i, j)\):

\[T_{2n}(i, 2j) = T_d(i, j)\]

and

\[T_{2n}(i, 2j + 1) = T_d(i, j)\]

for \(j \in \{0, n - 1\}\)

(b) the last \(n\) basis vectors of \(T_{2n}(i, j)\):

(i) \(T_{2n}(i + n, 2j) = T_d(i, j)\)

and

\[T_{2n}(i + n, 2j + 1) = - T_d(i, j)\]

for \(j \in \{0, 2, 4, \ldots, n - 2\}\)

(ii) \(T_{2n}(i + n, 2j) = - T_d(i, j)\)

and

\[T_{2n}(i + n, 2j + 1) = T_d(i, j)\]

for \(j \in \{1, 3, 5, \ldots, n - 1\}\)

It was found that the basis vectors of \(T_{2n}(i, j)\) generated using the above method resemble those of the \(T_d(i, j)\). The
order-8 ICT may therefore be generalised to any order-2^m ICT for m > 3. Computer programs have been used to search for order-16 ICTs whose transform efficiency is the highest for a limited to 7, 15, 31, up to 255 and e equal to 3 and f equal to 1. It was found that ICT(246, 222, 147, 50, 3, 1) has the highest transform efficiency for a less than or equal to 255 and ICT(10, 9, 6, 2, 3, 1) has the highest transform efficiency for a less than or equal to 128. However, as given by Table 8, the transform efficiency performance of the order-16 ICTs is inferior to that of the other well known order-16 orthogonal transforms. This could be because the order-16 ICTs have only eight levels in their kernel whilst the DCT and the slant transform have 16 levels.

7 Conclusion

In this paper, the concept of dyadic symmetry has been used to modify the order-8 DCT and generate many new integer orthogonal transforms which are called the integer cosine transforms (ICTs). The basis vectors of these transforms are similar to those of the DCT and all ICTs can be implemented by integer arithmetic. The implementation complexity of an ICT depends on the number of bits required to represent the magnitude of its kernel components. In comparison with the CMT, which is an integer approximation of the DCT, an ICT whose transform kernel contains only 10, 9, 6, 2, 3, and 1 has similar implementation complexity but higher decorrelation ability and better mean-square-error performance. Even better performance can be achieved by some ICTs whose kernel components require longer bit lengths for representation. ICTs that require three bits or less for representation of their component magnitudes are available but have a less satisfactory performance. The availability of many ICTs thus provides an engineer with the freedom to tradeoff performance for simple implementation in designing a transform codec. The order-8 ICT can be computed using a fast computational algorithm which is similar to that of the order-8 DCT but requires only integer multiplication and addition operations. The order-8 ICTs can be generalised to transforms of larger block sizes, however the performance cannot match that of the order-8 transforms.

8 References