ISO/IEC JTC1/SC29/WG1**N 1720**
26 June 2000

ISO/IEC JTC1/SC29/WG1
(ITU-T SG8)

Coding of Still Pictures

JBIG	**JPEG**
Joint Bi-level Image Experts Group	Joint Photographic Experts Group

TITLE:	**Proposal of reversible integer implementation for multiple component transforms**
SOURCE:	**Center for Information Science** **Peking University, Beijing, 100871, CHINA** **(phao@cis.pku.edu.cn)**
AUTHORS:	**Pengwei Hao, Qingyun Shi**
PROJECT:	**JPEG 2000**
STATUS:	
REQUESTED ACTION:	**For review at the July 2000 meeting of WG1.**
DISTRIBUTION:	**WG1 Web Pages**

Contact:
ISO/IEC JTC 1/SC 29/WG 1 Convener - Dr. Daniel T. Lee
Hewlett-Packard Company, 11000 Wolfe Road, MS42U0, Cupertino, California 95014, USA
Tel: +1 408 447 4160, Fax: +1 408 2842, E-mail: daniel_lee@hp.com

Abstract

An integer reversible implementation of multiple component transform is significant for lossless, unified lossy/lossless and progressive lossy-lossless multi-component or hyper-spectral image coding. In this proposal, we present a reversible integer mapping method for invertible linear transforms. Its effectiveness was evaluated using 2 6-band Landsat TM images and an 80-component hyper-spectral image. The linear transforms we tested and compared are KLT, DWT, and a direct prediction transform. Both lossy and lossless compression performance were considered, as well as the computational complexity. Experiments with JPEG-2000 verification model (VM7.2) showed that KLT or reversible KLT outperforms other approaches for all of the images in the case of lossy and lossless compression.

Introduction

Multiple component transform (MCT) is one key step in multi-component or hyper-spectral image coding. The integer reversible transform is especially useful in lossless compression, unified lossy/lossless image coding system, lossy-to-lossless progressive transmission, and lossy compression with the lossless reproduction of a region of interest [1]. In light of these applications, we can see that there is a clear need to consider some reversible integer implementation for known linear transforms which are good enough for data decorrelation. It is well-known that a wavelet transform can be factorized into a series of lifting steps which can be realized reversibly with integers [2]. We have also proved mathematically that any linear transform can be implemented by integer mapping [3]. In another paper accepted by ICPR 2000, [4], we found an optimal way to implement a linear transform of three components with integer mapping. The corresponding proposal to JPEG-2000 was numbered N1479, [5], but it is not satisfying when we used a perceptibly uniform scalar criterion to assess it later. At the beginning of year 2000, we proved that the necessary and sufficient condition for a matrix to be factorized into a series of **elementary reversible matrices** (ERMs) is that the absolute value of the matrix determinant is 1[6]. And we also find some efficient ways to the factorizations, **triangular ERM** (TERM) and **single-row ERM** (SERM) factorization.

In this proposal, we presented an approach to factorize a transform matrix into SERMs, and applied the method to multiple component transforms in JPEG-2000. Its effectiveness was evaluated using 2 6-band Landsat TM images and an 80-component hyper-spectral image. The linear transforms we tested and compared are KLT, DWT, a direct prediction transform similar to RCT [7] adopted by JPEG-2000, and a tasselled cap transform for TM satellite images only.

Both lossy and lossless compression performances were considered, as well as the computational complexity. Experiments with JPEG-2000 verification model (VM7.2) showed that KLT or reversible KLT outperforms other approaches for almost all of the images in the case of lossy and lossless compression. The fixed tasselled cap transform performs only a little worse than the best for TM data.

This work is also an extension of the past work described by Wilkinson and Kasner in [8].

Multiple Component Transforms

Besides discrete wavelet transform (DWT), there are a lot of linear transforms which can be applied for image data decorrelation such as a direct prediction transform (DPT), Karhunen-Loeve transform (KLT). For TM satellite images, there is another transform found by Crist and Cicone to extract the information of luminance, green and moisture from infrared reflectance images, which is called tasselled cap transform (TCT).

KLT is a transform depending upon the data to be transformed. Its matrix consists of the eigenvectors derived from of the covariance matrix of all the data, so a KLT matrix is an orthogonal matrix, whose determinant is ±1 and inverse is its transpose. From a viewpoint of least mean error, KLT is the best transform for data decorrelation.

For TM remote sensing images, a TCT matrix was given by Crist and Cicone:

$$T = \begin{bmatrix} 0.3037 & 0.2793 & 0.4743 & 0.5585 & 0.5082 & 0.1863 \\ -0.2848 & -0.2435 & -0.5436 & 0.7243 & 0.0840 & -0.1800 \\ 0.1509 & 0.1973 & 0.3279 & 0.3406 & -0.7112 & -0.4572 \\ -0.8242 & 0.0849 & 0.4392 & -0.0580 & 0.2012 & -0.2768 \\ -0.3280 & 0.0549 & 0.1075 & 0.1855 & -0.4357 & 0.8085 \\ 0.1084 & -0.9022 & 0.4120 & 0.0573 & -0.0251 & 0.0238 \end{bmatrix}$$

It is also an orthogonal matrix.

The reversible color transform, RCT, adopted by JPEG-2000, made excellent performance, which is defined [7]:

$$\text{RCT: } \begin{cases} Y = \left[\dfrac{R+2G+B}{4}\right] \\ Cr = R - G \\ Cb = B - G \end{cases}, \text{ and its inverse: } \begin{cases} G = Y - \left[\dfrac{Cr+Cb}{4}\right] \\ R = Cr + G \\ B = Cb + G \end{cases}$$

Or in matrix form, the corresponding transform matrix is:

$$\begin{bmatrix} Y \\ Cr \\ Cb \end{bmatrix} = \begin{bmatrix} 1/4 & 1/2 & 1/4 \\ 1 & -1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Owing to the inspiration, we can also construct a similar matrix for N points:

$$T = \begin{bmatrix} a_1 & a_2 & \cdots & a_{N-1} & a_N \\ 1 & -1 & & & \\ & \cdots & \cdots & & \\ & & 1 & -1 & \\ & & & 1 & -1 \end{bmatrix}$$

where $\sum_{n=1}^{N} a_n = 1$ and we have $\det(T) = \pm 1$.

For each of its lower N-1 rows are a direct prediction with the following element, the transform can also be called as a direct prediction transform (DPT). An advantage of this transform is that the dynamic range of the transformed values are fixed and the transform computation is simple.

Of course, we can also define a lot of simple transforms, such as Gram-Schmidt transform matrix, other linear prediction transforms.

The wavelet filter employed as MCT in the experiments is Cohen-Daubechies-Feauveau bi-orthogonal 5/3 filter implemented by lifting scheme for both lossy and lossless compression.

Reversible Integer Mapping

For a linear transform that maps an integer number x to an integer number y, $y=ax+b$, the necessary and sufficient condition for reversible mapping is that $|a| = 1$, and b is an integer number. We named a number of $|a| = 1$ as an **integer factor**, which is denoted as j, and its reciprocal $j'=1/j$. For real number mapping, the condition is $a = \pm 1$ and using a rounding arithmetic for b, $[b]$. Where [ ] is an arithmetic rounding operation(rounding or chopping or some bits before decimal point). The flow chart of the transform and its inverse can be illustrated as in Fig. 1.

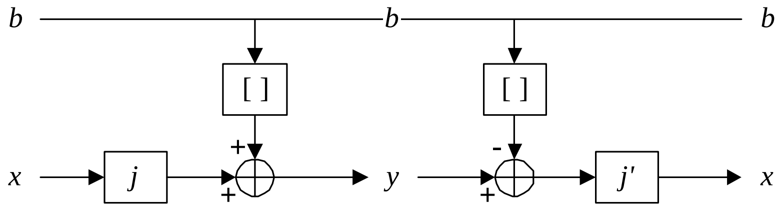

Fig. 1 Forward and reverse linear transform of a number implemented by integer mapping

In the case of a=1, or j=1, the structure is the same as the ladder structure proposed by Bruekers & van den Enden (1992)[9] and the lifting scheme structure by Daubechies & Sweldens (1996)[2].

We proved that any N-by-N square matrix $\boldsymbol{A}$ whose determinant is ± 1 can be factorized into a product of N+1 SERMs. A unit SERM can be described as $\boldsymbol{S}_m = \boldsymbol{I} + \boldsymbol{e}_m \boldsymbol{s}_m^T$, where $\boldsymbol{e}_m$ is the m-th standard basis vector formed as the m-th column of the identity matrix, $\boldsymbol{s}_m = \{s_{mn}\}(n = 1,2,\cdots,N)$ is a vector that the m-th element, s_{mm}, is 0. The unit SERMs share an important property: $\boldsymbol{S}_m^{-1} = \boldsymbol{I} - \boldsymbol{e}_m \boldsymbol{s}_m^T$. For convenience, we denote $\boldsymbol{S}_0 = \boldsymbol{I} + \boldsymbol{e}_N \boldsymbol{s}_0^T$ in our following discussion. So the SERM factorization can be $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{S}_N \boldsymbol{S}_{N-1} \cdots \boldsymbol{S}_1 \boldsymbol{S}_0 \boldsymbol{J}$, where $\boldsymbol{P}$ is a permutation matrix for partial pivoting, $\boldsymbol{J}$ is a diagonal matrix whose diagonal elements are all integer factors. If complete pivoting is applied, the SERM factorization can also be $\boldsymbol{A} = \boldsymbol{P}_L \boldsymbol{S}_N \boldsymbol{S}_{N-1} \cdots \boldsymbol{S}_1 \boldsymbol{S}_0 \boldsymbol{P}_R \boldsymbol{J}$. For each SERM, $\boldsymbol{S}_m$, its corresponding transform can be implemented in-place for m-th element only : $y_m = x_m + \left[\sum_{n \neq m} s_{mn} x_n\right] = x_m + [b]$ where $m = 1,2,3,\cdots,N$

The implementation with N +1 SERMs has a flowchart structure as Fig. 2 illustrated.

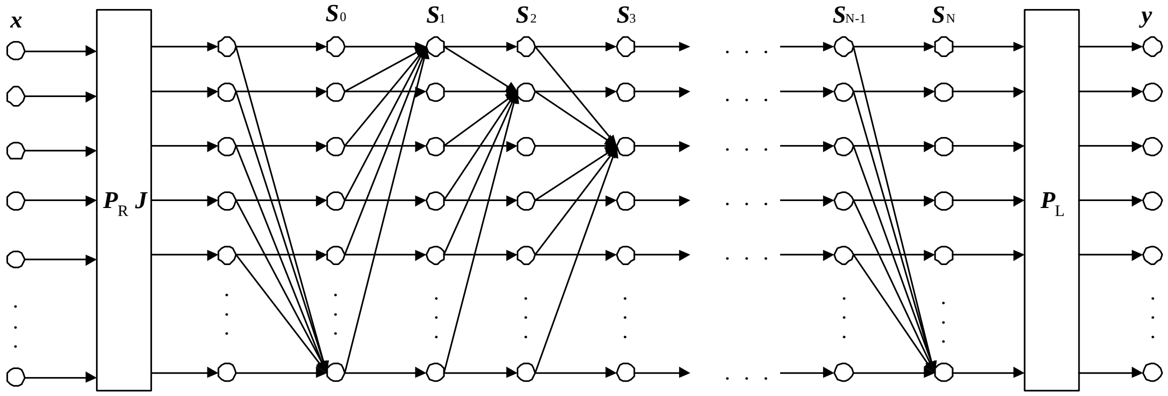

Fig. 2 A flowchart structure of linear transform implemented by SERMs

Matrix Factorization

It has been proved that a nonsingular matrix can be factorized into a product of 3 TERMs or a series of SERMs. An algorithm of matrix factorization for reversible integer mapping is given hereinbelow.

Suppose

$$\boldsymbol{A}=\begin{bmatrix} a_{1,1}^{(0)} & a_{1,2}^{(0)} & \cdots & a_{1,N}^{(0)} \\ a_{2,1}^{(0)} & a_{2,2}^{(0)} & \cdots & a_{2,N}^{(0)} \\ \cdots & \cdots & \cdots & \cdots \\ a_{N,1}^{(0)} & a_{N,2}^{(0)} & \cdots & a_{NN}^{(0)} \end{bmatrix}$$

There must exist a permutation matrix $\boldsymbol{P}_1$ for row interchanges, such that:

$$\boldsymbol{P}_1\boldsymbol{A}=\begin{bmatrix} p_{1,1}^{(1)} & p_{1,2}^{(1)} & \cdots & p_{1,N}^{(1)} \\ p_{2,1}^{(1)} & p_{2,2}^{(1)} & \cdots & p_{2,N}^{(1)} \\ \cdots & \cdots & \cdots & \cdots \\ p_{N,1}^{(1)} & p_{N,2}^{(1)} & \cdots & p_{NN}^{(1)} \end{bmatrix}$$

and $p_{1,N}^{(1)} \neq 0$, and hence there must be a number s_1, such that $p_{1,1}^{(1)} - s_1 \cdot p_{1,N}^{(1)} = 1$. Then, we can get $s_1 = \left(p_{1,1}^{(1)} - 1\right) / p_{1,N}^{(1)}$, and obtain a product of

$$\boldsymbol{P}_1\boldsymbol{A}\boldsymbol{S}_{0,1} = \boldsymbol{P}_1\boldsymbol{A}\begin{bmatrix} 1 & & \\ & \boldsymbol{I} & \\ -s_1 & \boldsymbol{0} & 1 \end{bmatrix} = \begin{bmatrix} 1 & p_{1,2}^{(1)} & \cdots & p_{1,N}^{(1)} \\ p_{2,1}^{(1)} - s_1 p_{2,N}^{(1)} & p_{2,2}^{(1)} & \cdots & p_{2,N}^{(1)} \\ \cdots & \cdots & \cdots & \cdots \\ p_{N,1}^{(1)} - s_1 p_{N,N}^{(1)} & p_{N,2}^{(1)} & \cdots & p_{NN}^{(1)} \end{bmatrix}$$

Then the forward elimination of the first column is just to multiply an elementary Gauss matrix $\boldsymbol{L}_1$:

$$\boldsymbol{L}_1\boldsymbol{P}_1\boldsymbol{A}\boldsymbol{S}_{0,1} = \begin{bmatrix} 1 & & \\ s_1 p_{2,N}^{(1)} - p_{2,1}^{(1)} & 1 & \\ \cdots & & \boldsymbol{I} \\ s_1 p_{N,N}^{(1)} - p_{N,1}^{(1)} & & 1 \end{bmatrix} \boldsymbol{P}_1\boldsymbol{A}\boldsymbol{S}_{0,1} = \begin{bmatrix} 1 & a_{1,2}^{(2)} & \cdots & a_{1,N}^{(2)} \\ 0 & a_{2,2}^{(2)} & \cdots & a_{2,N}^{(2)} \\ \cdots & \cdots & \cdots & \cdots \\ 0 & a_{N,2}^{(2)} & \cdots & a_{NN}^{(2)} \end{bmatrix}$$

Continuing in this way, for $k = 2,3,\cdots,N-1$, $\boldsymbol{P}_k$ define the row interchanges among the *k*-th through the *N*-th rows so as to guarantee the *k*-th element in the *N*-th column not to be zero, $p_{k,N}^{(k)} \neq 0$. (If there were no such element, $\boldsymbol{A}$ should have been singular.) $\boldsymbol{S}_{0,k}$ covert $a_{k,k}^{(k)}$ into 1's, where $s_k = \left(p_{k,k}^{(k)} - 1\right) / p_{k,N}^{(k)}$. And $\boldsymbol{L}_k$ record the row multipliers used for elimination of column *k*. Then we get,

$$\boldsymbol{L}_{N-1}\boldsymbol{P}_{N-1}\cdots\boldsymbol{L}_2\boldsymbol{P}_2\boldsymbol{L}_1\boldsymbol{P}_1\boldsymbol{A}\boldsymbol{S}_{0,1}\boldsymbol{S}_{0,2}\cdots\boldsymbol{S}_{0,N-1} = \begin{bmatrix} 1 & a_{1,2}^{(N-1)} & \cdots & a_{1,N}^{(N-1)} \\ 0 & 1 & \cdots & a_{2,N}^{(N-1)} \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & a_{N,N}^{(N-1)} \end{bmatrix} = \boldsymbol{U}\boldsymbol{D}_R$$

where $\boldsymbol{D}_R = diag\,(1,1,\cdots,1,a_{N,N}^{(N-1)})$, and

$$\boldsymbol{U} = \begin{bmatrix} 1 & a_{1,2}^{(N-1)} & \cdots & a_{1,N}^{(N-1)} / a_{N,N}^{(N-1)} \\ 0 & 1 & \cdots & a_{2,N}^{(N-1)} / a_{N,N}^{(N-1)} \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

Having multiplied all the SERMs together, all the permutation matrices together, and all the unit lower triangular matrices together, respectively, we have one SERM, one premultiplying permutation matrix, and one unit lower triangular matrix with the rows sorted into their final ordering. Since the inverse of a unit lower triangular matrix is also a unit lower triangular matrix, and the transpose of a permutation matrix is also a permutation matrix, we have

$$\boldsymbol{S}_{0,1}\boldsymbol{S}_{0,2}\cdots\boldsymbol{S}_{0,N-1}=\begin{bmatrix} 1 & & & \\ & \boldsymbol{I} & & \\ & & 1 & \\ -s_1 & \cdots & -s_{N-1} & 1 \end{bmatrix}=\boldsymbol{S}_{0R}^{-1}$$

$$\boldsymbol{L}_{N-1}\boldsymbol{P}_{N-1}\cdots\boldsymbol{L}_2\boldsymbol{P}_2\boldsymbol{L}_1\boldsymbol{P}_1=\boldsymbol{L}_{N-1}\left(\boldsymbol{P}_{N-1}\boldsymbol{L}_{N-2}\boldsymbol{P}_{N-1}^T\right)\cdots\cdots\left(\boldsymbol{P}_{N-1}\cdots\boldsymbol{P}_2\boldsymbol{L}_1\boldsymbol{P}_2^T\cdots\boldsymbol{P}_{N-1}^T\right)\left(\boldsymbol{P}_{N-1}\cdots\boldsymbol{P}_2\boldsymbol{P}_1\right)=\boldsymbol{L}^{-1}\boldsymbol{P}^T$$

Hence, we obtain $\boldsymbol{L}^{-1}\boldsymbol{P}^T\boldsymbol{A}\boldsymbol{S}_{0R}^{-1}=\boldsymbol{U}\boldsymbol{D}_R$ or $\boldsymbol{A}=\boldsymbol{P}\boldsymbol{L}\boldsymbol{U}\boldsymbol{D}_R\boldsymbol{S}_{0R}$.

For real matrix $\boldsymbol{A}$, if $\det\boldsymbol{A}=\pm1$, then $a_{N,N}^{(N-1)}=\pm1$ is an integer factor, and the TERM factor matrices can also be real. Let $\boldsymbol{J}=\boldsymbol{D}_R$, $\boldsymbol{J}\boldsymbol{S}_{0R}=\boldsymbol{S}_0\boldsymbol{J}$, we have $\boldsymbol{A}=\boldsymbol{P}\boldsymbol{L}\boldsymbol{U}\boldsymbol{S}_0\boldsymbol{J}$.

Once having got a TERM factorization, we can obtain the corresponding SERM factorization easily. Following factorization can be carried through:

$$\boldsymbol{L}\boldsymbol{U}=\left(\boldsymbol{L}\boldsymbol{U}\boldsymbol{S}_1^{-1}\right)\boldsymbol{S}_1=\left(\boldsymbol{L}\boldsymbol{U}\boldsymbol{S}_1^{-1}\boldsymbol{S}_2^{-1}\right)\boldsymbol{S}_2\boldsymbol{S}_1=\cdots=\left(\boldsymbol{L}\boldsymbol{U}\boldsymbol{S}_1^{-1}\boldsymbol{S}_2^{-1}\cdots\boldsymbol{S}_{N-1}^{-1}\right)\boldsymbol{S}_{N-1}\cdots\boldsymbol{S}_2\boldsymbol{S}_1=\boldsymbol{S}_N\cdots\boldsymbol{S}_2\boldsymbol{S}_1$$

where the m-th row of $\boldsymbol{S}_m$ equals to the m-th row of $\boldsymbol{L}\boldsymbol{U}\boldsymbol{S}_1^{-1}\boldsymbol{S}_2^{-1}\cdots\boldsymbol{S}_{m-1}^{-1}$, $m=1,2,3,\cdots,N$.

Thus, we can obtain a SERM factorization as : $\boldsymbol{A}=\boldsymbol{P}\boldsymbol{S}_N\boldsymbol{S}_{N-1}\cdots\boldsymbol{S}_1\boldsymbol{S}_0\boldsymbol{J}$

For the DPT defined in the second section, we can permute the matrix and factorize it into reversible DPT (RDPT) matrices as:

$$\begin{aligned}
T&=\begin{bmatrix} & & & & 1\\ 1 & & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & \end{bmatrix}\cdot\begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \cdots & \cdots & \\ & & & 1 & -1\\ \boldsymbol{a}_1 & \boldsymbol{a}_2 & \cdots & \boldsymbol{a}_{N-1} & \boldsymbol{a}_N \end{bmatrix}\\
&=\begin{bmatrix} & & & & 1\\ 1 & & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & \end{bmatrix}\cdot\begin{bmatrix} 1 & & & & \\ & 1 & -1 & & \\ & & \cdots & \cdots & \\ & & & 1 & -1\\ \boldsymbol{a}_1 & \boldsymbol{a}_1+\boldsymbol{a}_2 & \cdots & \boldsymbol{a}_{N-1} & \boldsymbol{a}_N \end{bmatrix}\cdot\begin{bmatrix} 1 & -1 & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix}\\
&=\begin{bmatrix} & & & & 1\\ 1 & & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & \end{bmatrix}\cdot\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \cdots & \cdots & \\ & & & 1 & -1\\ \boldsymbol{a}_1 & \boldsymbol{a}_1+\boldsymbol{a}_2 & \cdots & \boldsymbol{a}_{N-1} & \boldsymbol{a}_N \end{bmatrix}\cdot\begin{bmatrix} 1 & & & & \\ & 1 & -1 & & \\ & & \cdots & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix}\cdot\begin{bmatrix} 1 & -1 & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix}\\
&=\cdots\cdots\cdots\\
&=\begin{bmatrix} & & & & 1\\ 1 & & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & \end{bmatrix}\cdot\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \cdots & \cdots & \\ & & & 1 & \\ \boldsymbol{a}_1 & \boldsymbol{a}_1+\boldsymbol{a}_2 & \cdots & \boldsymbol{a}_1+\boldsymbol{a}_2+\cdots+\boldsymbol{a}_{N-1} & \boldsymbol{a}_1+\boldsymbol{a}_2+\cdots+\boldsymbol{a}_N \end{bmatrix}\cdot\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & -1\\ & & & & 1 \end{bmatrix}\cdots\cdots\begin{bmatrix} 1 & -1 & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix}\\
&=\begin{bmatrix} & & & & 1\\ 1 & & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & \end{bmatrix}\cdot\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \cdots & \cdots & \\ & & & 1 & \\ \boldsymbol{a}_1 & \boldsymbol{a}_1+\boldsymbol{a}_2 & \cdots & \boldsymbol{a}_1+\boldsymbol{a}_2+\cdots+\boldsymbol{a}_{N-1} & 1 \end{bmatrix}\cdot\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & -1\\ & & & & 1 \end{bmatrix}\cdots\cdots\begin{bmatrix} 1 & -1 & & & \\ & 1 & & & \\ & & \cdots & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix}\\
&=\boldsymbol{P}\boldsymbol{S}_N\boldsymbol{S}_{N-1}\cdots\boldsymbol{S}_2\boldsymbol{S}_1
\end{aligned}$$

In the factorization, a clear advantage of RDPT is that only the last SERM transform needs float computations. If all a_k are rational numbers with a denominator of 2^n, the final calculation can also be implemented by integer multiplications and shifts only.

Evaluation Methodology

For evaluation purposes, version 7.2 of the JPEG-2000 verification model (VM) software was employed. In order to compare the multiple component transforms equally, we used the VM as an encoder without any third dimension transform (parameter '-Mtdt N'). We applied an MCT and its inverse before and after VM. The test strategy is shown in Fig. 3.

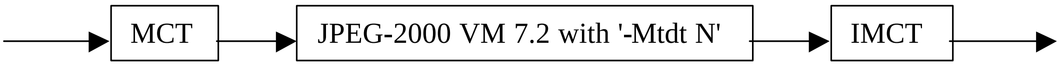

Fig. 3 Test strategy of multiple component transforms

MCTs are no transform (NoT), DWT, reversible DPT (RDPT), reversible tasselled cap transform (RTCT), KLT, and reversible KLT (RKLT). The wavelet filter employed in the experiments is Cohen-Daubechies-Feauveau bi-orthogonal 5/3 filter for lossy and lossless compression.

The test data used in our study consisted of the images listed in Table 1.

Table 1: Test Images

Index	Image	Size	Bands	Description
1	Beijing	512x512	6	Urban areas of Beijing
2	Powerplant	512x512	6	A power plant located in west to Beijing city
3	XB	346x512	80	Road, water, rice paddy

Images of `Beijing` and `Powerplant` are two sub-images of a Landsat TM image, consist of 6 bands of the same resolution, namely band 1, 2, 3, 4, 5 and 7. Image `XB` is a high-resolution and hyperspectral image acquired by PHI(Pushbroom Hyperspectral Imager), radiometric corrected. It was contributed by Institute of Remote Sensing Applications, Chinese Academy of Sciences, China.

Using the transforms under consideration, each of the test images was compressed in a lossy manner at several bit rates, 0.5 bits per pixel per band (bpppb), 0.25 bpppb, 0.125 bpppb, and 0.0625 bpppb, in addition to being compressed losslessly. For lossless compression, the final bit rate measured by bits per pixel per band (bpppb) was used to evaluate the performance. In the lossy compression case, objective image distortion metrics, PSNR, was used instead:

$$PSNR = 10\log_{10}\frac{N\cdot 255^2}{\sum_{k=1}^{N} MSE(B_k)}$$

In order to display the contrast between the direct KLT and its integer reversible implementation, we also list the data with the direct transform, and lossless compression performance of the residual (eKLT) and the total bit rates.

Lossless and Lossy Compression Performance

Image quality test was undertaken as discussed in the previous section. The test results of three multiple component images are shown in Tables 2 and 3. The relatively best performance is highlighted with bold digits.

Each of the test images listed in Table 1 was compressed in a lossless manner using the MCTs, and the results are shown in Table 2.

Table 2: Lossless compression performance for multiple component test images (Bit Rate, bpppb)

Image	NOT	DWT	RDPT	RTCT	RKLT	KLT	eKLT	KLT+eKLT
Beijing	4.29	4.04	4.20	3.93	**3.77**	3.70	1.61	5.31
Powerplant	3.98	3.84	4.04	3.75	**3.67**	3.57	1.39	4.96

XB	5.51	4.95	5.18	- -	**4.89**	4.84	1.54	6.38

Through the results we can see that RKLT has the best performance in the case of lossless compression. The fixed transform RTCT is the second best for TM images.

Each of the test images listed in Table 1 was also compressed in a lossy manner using the MCTs, and the results are shown in Table 3.

Table 3: Lossy compression results for multiple component test images (PSNR, dB)

Bit Rate (BPPPB)	Image	Multiple Component Transform					
		NoT	DWT	RDPT	RTCT	RKLT	KLT
0.0625	Beijing	32.14	32.70	31.36	32.69	**33.23**	33.12
	Powerplant	32.74	33.18	31.36	33.32	**33.64**	33.45
	XB	19.27	25.03	13.80	- -	**26.70**	26.64
0.125	Beijing	33.15	33.76	32.09	33.89	**34.63**	34.50
	Powerplant	34.03	34.52	32.55	34.74	**35.07**	34.82
	XB	21.38	26.75	15.24	- -	**28.31**	28.24
0.25	Beijing	34.47	35.38	33.51	35.61	**36.57**	36.44
	Powerplant	35.67	36.34	34.11	36.67	**37.12**	36.80
	XB	23.96	28.96	17.51	- -	**30.30**	30.21
0.5	Beijing	36.27	37.46	34.58	37.75	**38.99**	38.86
	Powerplant	37.73	38.54	35.47	39.01	**39.68**	39.16
	XB	27.27	31.68	19.93	- -	**32.69**	32.58

All of the results share one conclusion: RKLT is the best overwhelmingly, and RTCT is a little worse than RKLT for TM images, but still excellent.

Computational Complexity

The integer reversible transforms are calculated using float operations, but rounding arithmetic and in-place integer assignations. Clearly, reversible implementation introduces no more additional FLOPS than its original direct linear transform. Moreover, every transform built with SERMs is immediately invertible, and its inverse has exactly the same computational complexity as the forward transform.

For an N-by-N linear transform matrix, the computational complexity of each reversible transform is given in Table 4.

Table 4: Computational complexity comparison

Implementation	Addition	Multiplication	Rounding	Permutation
DWT(5/3, 4 levels)	$15N/4$	$15N/8$	$15N/8$	Unnecessary
RDPT	$2(N-1)$	$N-1$	N	Unnecessary
KLT	N^2-N	N^2	N	No
RKLT	N^2-1	N^2-1	$N+1$	Needed

Conclusions

Our proposed integer reversible implementation method of multiple component transform outperforms the original linear transform calculations. It gives a possibility to apply a lossless transform in image coding. In our experiments, the reversible KLT has acceptable computational complexity and gives the best performance when used for both lossless and lossy compression. A fixed transform, TCT, is also excellent for TM images, so it is very possible to find one proper transform and its integer implementation for a very class of multiple component images.

Acknowledgements

The authors would like to express their appreciation to supporter of NKBRSF grant G1998030606, their gratitude to Dr. Prof. Jin Li of Microsoft Research China for all of his help, and thanks to the Institute of Remote Sensing Applications, Chinese Academy of Sciences, China for their providing the test images.

References

[1] Report on Core Experiment CodEff4. ISO/IEC JTC1/SC29/WG1 N1015, Oct. 1998

[2] I. Daubechies, W. Sweldens, Factoring wavelet transforms into lifting steps, J. of Fourier Analysis and Application, 1998, V4, N3, pp 247-269.

[3] P. Hao and Q. Y. Shi, “Invertible Linear Transforms Implemented by Integer Mapping” (in Chinese), Science in China, April 2000, V30, N2, pp132-141.

[4] P. Hao and Q.Y. Shi, “Comparative Study of Color Transforms for Image Coding and A Derivation of Integer Reversible Color Transform”, accepted by ICPR2000, Sept. 3-8, 2000, Barcelona, Spain.

[5] P. Hao, Q.Y. Shi and G. Xing, “Proposal of an integer reversible color transform”, ISO/IEC JTC1/SC29/WG1N 1479, Nov. 1999.

[6] P. Hao and Q.Y. Shi, “Matrix Factorizations for Reversible Integer Mapping”, submitted to IEEE Transactions on Signal Processing, March 2000.

[7] M. J. Gormish, E. L. Schwartz, A. Keith, M. Boliek and A. Zandi, “Lossless and nearly lossless compression for high quality images”, Proceedings of the SPIE/IS&T Conference on Very High Resolution and Imaging II, vol. 3025, February 1997, San Jose, CA, pp. 62-70.

[8] T. S. Wilkinson and J. H. Kasner, “Lossless Hyperspectral Coding in JPEG 2000”, ISO/IEC JTC1/SC29/WG1, N1631,Tokyo, Japan, March 2000.

[9] F. A. M. L. Bruekers, A. W. M. van den Enden, New networks for perfect inversion and perfect reconstruction, IEEE J. on Selected Areas in Communications, 1992, V10, N1, 130-137.