A Bit Planes Predictive-Transform Source Coder Illustrated with SAR Imagery

Erlan H. Feria and Dalibor Licul
The College of Staten Island (CSI) of the City University of New York (CUNY)
Department of Engineering Science and Physics
2800 Victory Blvd, Staten Island, New York, 10314
feria@mail.csi.cuny.edu

ABSTRACT

The memory space compression of a predictive-transform (PT) source coder is found to remain outstanding when the quantized coefficient errors emanating from its 'lossy' PT source encoder section are encoded using a novel, fast, and simple bit planes methodology. The advanced technique outperforms wavelets based JPEG2000 by more than 5 dBs when it compresses by a factor of 8,192 a test 4 megabytes (MB) synthetic aperture radar (SAR) image used in knowledge-aided airborne moving target indicator (AMTI) radar.

Keywords: Source coding, memory space compression, SAR imagery, predictive-transform, radar-blind, bit planes, lossy, lossless, AMTI

1. INTRODUCTION

The theory and practice of source coding has a renowned recent history and is an enabling technology for what is known today as the information revolution [1]. Source coding deals with the memory space compression of signals emanating from a signal source. Each of the possible signal source output outcomes conveys some amount of information which is measured by the logarithm of the reciprocal of the probability of the outcome. As a result an unlikely outcome provides a large amount of information while one that often occurs does not. When the logarithm has base two the information is given in units of bits. The expected outcome information is called the signal source entropy and is given in units of bits per outcome. This entropy is often significantly smaller than the signal source rate and can then be used as a guide in designing a source coder. A fundamental problem in source coding is then to find a replacement for the signal source, called a source coder, characterized by a rate that emulates the signal source entropy. This type of source coder is 'lossless' since its output is the same as that of the signal source such as is the case with Huffman, Entropy, and Arithmetic coders. Another essential problem in source coding pertains to the design of lossy source coders that achieve rates that are significantly smaller than the signal source entropy. These solutions are linked to applications where the local signal to noise ratio (SNR) of the source coder does not have to be infinite or alternately the global performance criterion of the application at hand is not the local SNR. An example of the last case is when synthetic aperture radar (SAR) imagery is compressed for use in knowledge-aided (KA) airborne moving target indicator (AMTI) radar [2]. To address the 'lossy' source coding problem many techniques have been developed including the standards of JPEG, MPEG, wavelets based JPEG2000 [3], and predictive-transform (PT) source coding [4]. In this paper it will be shown that a very fast and simple bit planes based PT source coder significantly outperforms wavelets based JPEG2000 when applied to SAR imagery.

PT source coding naturally arises from the unification of predictive source coding with transform source coding with the goal of trading off the implementation simplicity of a predictive source coder with the high speed of a

This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) under the KASSPER Program Grant No. FA8750-04-1-004.

Mobile Multimedia/Image Processing for Military and Security Applications, edited by Sos S. Agaian,
Sabah A. Jassim, Proc. of SPIE Vol. 6250, 625004, (2006) · 0277-786X/06/$15 · doi: 10.1117/12.666154

transform coder. This unification is characterized by a source coder architecture with 'fixed' prediction and transformation matrices that are derived 'off-line' from coupled Wiener-Hopf and eigensystem design equations using stationary signal statistics [5]. These design equations result when a mean squared error (MSE) criterion subjected to appropriate quantizer constraints is minimized with respect to the prediction and transformation matrices. Furthermore it has been determined [5] that simplifying decomposed PT structures arise when signals are symmetrically processed. A strip processor, used later to illustrate the scheme developed in this paper, is an example of such processing. Also cascaded Hadamard structures [4] have been integrated with PT structures to accelerate the on-line evaluation of the necessary products between the transform/ predictor matrix and a signal vector as well as the off-line evaluation of the transform and predictor matrices from the coupled Wiener-Hopf and eigensystem design equations. In this paper, it will be established that the excellent memory space compression achieved with PT source coding is not affected by its integration with a very fast and simple bit planes methodology that operates on the quantized coefficient errors emanating from the lossy PT encoder section. The efficacy of the approach will be illustrated by compressing by a factor of 8,192 a test 4 megabytes (MB) SAR image used in KA-AMTI radar that is subjected to severely taxing environmental disturbances. In particular it will be found that PT source coding with bit planes significantly outperforms wavelets based JPEG2000 in terms of local SNR as well as global SINR radar performance.

The organization of this paper is as follow. Section II provides the pre-requisite PT source coding background material. In Section III the proposed integration of PT source coding with bit planes is advanced. In Section IV simulation results are shown for a real-world application that clearly show that the advanced bit planes PT source coder is superior to wavelets based JPEG2000.

2. BACKGROUND

In Fig. 1 the global PT source coder architecture is shown. It has as its input the output of a signal source **y**. As an illustration this output will be assumed to be a real matrix representing 2-D images. The structure consists of two distinct sections. In the upper section the lossy encoder and associated lossy decoder are depicted while in the lower section the lossless encoder and decoder are shown. Before the lossless section of the coder is explained, which contains the offered bit planes, the lossy section will be reviewed. In Fig. 2 the lossy PT encoder structure is shown. It consists of a transform pre-processor $f_T(\mathbf{y})$ whose output $\mathbf{x}_k$ is a real n dimensional column vector. In Fig. 3 an image coding example

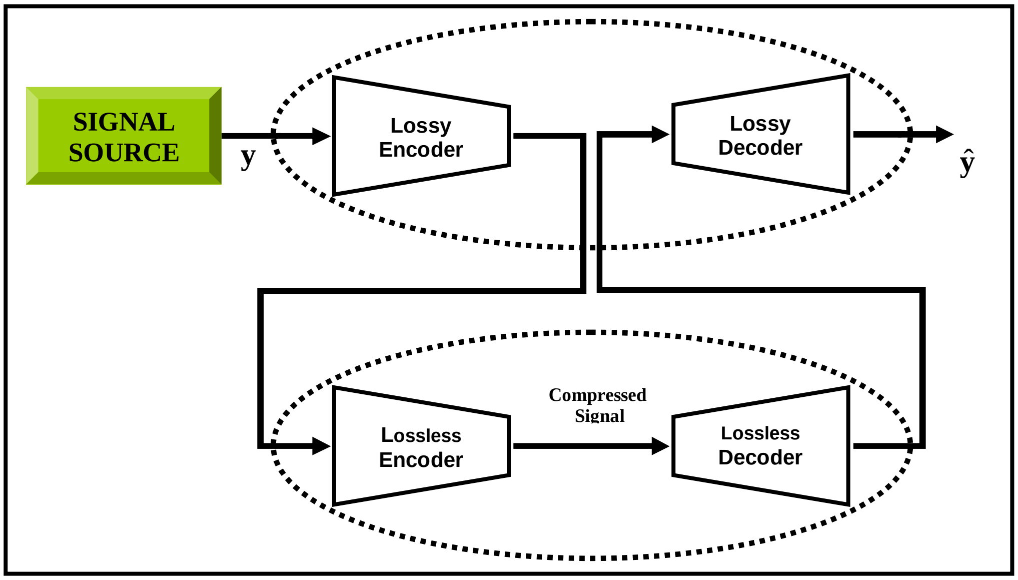


Fig. 1 The Global PT Source Coder Architecture

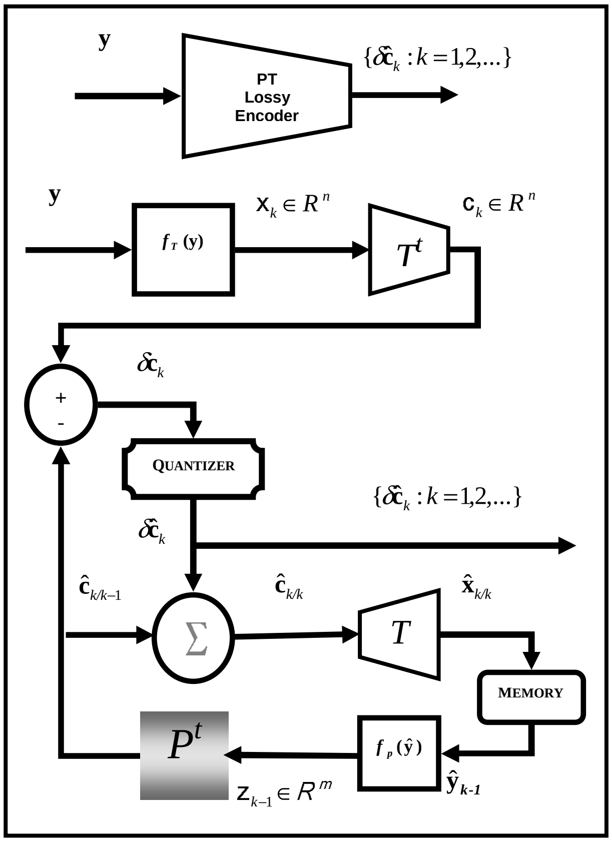
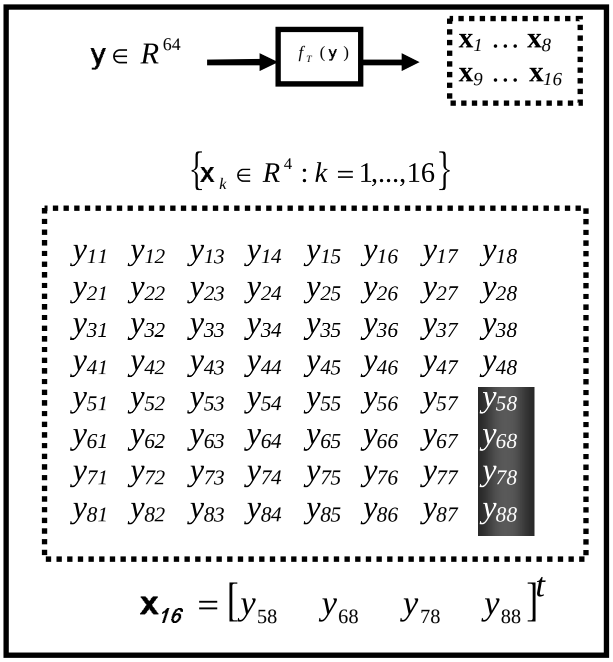


Fig. 2 Lossy PT Encoder Structure


$\mathbf{y} \in R^{64}$
$f_T(\mathbf{y})$
$\mathbf{x}_1 \ldots \mathbf{x}_8$
$\mathbf{x}_9 \ldots \mathbf{x}_{16}$
$\{\mathbf{x}_k \in R^4 : k=1,...,16\}$

$$\begin{array}{cccccccc}
y_{11} & y_{12} & y_{13} & y_{14} & y_{15} & y_{16} & y_{17} & y_{18}\\
y_{21} & y_{22} & y_{23} & y_{24} & y_{25} & y_{26} & y_{27} & y_{28}\\
y_{31} & y_{32} & y_{33} & y_{34} & y_{35} & y_{36} & y_{37} & y_{38}\\
y_{41} & y_{42} & y_{43} & y_{44} & y_{45} & y_{46} & y_{47} & y_{48}\\
y_{51} & y_{52} & y_{53} & y_{54} & y_{55} & y_{56} & y_{57} & y_{58}\\
y_{61} & y_{62} & y_{63} & y_{64} & y_{65} & y_{66} & y_{67} & y_{68}\\
y_{71} & y_{72} & y_{73} & y_{74} & y_{75} & y_{76} & y_{77} & y_{78}\\
y_{81} & y_{82} & y_{83} & y_{84} & y_{85} & y_{86} & y_{87} & y_{88}
\end{array}$$

$$\mathbf{x}_{16} = [y_{58} \quad y_{68} \quad y_{78} \quad y_{88}]^t$$

Fig. 3 Image Coding Illustration Transform Pre-Processing

is given where **y** is a matrix consisting of 64 real valued picture elements or pixels and the transform pre-processor produces sixteen n=4 dimensional pixel vectors {$\mathbf{x}_k$:k=1,…,16}. The pixel vector $\mathbf{x}_k$ then becomes the input of a nxn dimensional unitary transform matrix T. The multiplication of the transform matrix T by the pixel vector $\mathbf{x}_k$ produces an n dimensional real valued coefficient column vector $\mathbf{c}_k$. This coefficient, in turn, is predicted by a real n dimensional vector $\hat{\mathbf{c}}_{k/k-1}$. The prediction vector $\hat{\mathbf{c}}_{k/k-1}$ is derived by multiplying the real m dimensional output $\mathbf{z}_{k-1}$ of a predictor pre-processor (constructed using previously encoded pixel vectors as will be seen shortly), by a m x n dimensional real prediction matrix P. A real n dimensional coefficient error $\delta\mathbf{c}_k$ is then formed and subsequently quantized yielding $\delta\hat{\mathbf{c}}_k$. The quantizer has two assumed structures. One is an 'analog' structure that is used to derive analytical design expressions for the P and T matrices and another is a 'digital' structure used in actual compression applications. The analog structure consists of allowing the most energetic elements of $\delta\mathbf{c}_k$, say d of them, to pass to the quantizer output unaffected and the remaining elements to appear at the quantizer output as zero values, i.e.,

$$\delta\hat{\mathbf{c}}_k(i) = \begin{cases} \delta\mathbf{c}_k(i) & i=1,...,d \\ 0 & i=d+1,...n \end{cases}. \qquad (2.1)$$

The digital structure consists of multiplying $\delta\mathbf{c}_k$ by a real and scalar compression factor 'g' and then finding the closest integer representation for this real valued product, i.e.,

$$\delta\hat{\mathbf{c}}_k = \lfloor g\delta\mathbf{c}_k + 1/2 \rfloor. \qquad (2.2)$$

The quantizer output $\delta\hat{\mathbf{c}}_k$ is then added to the prediction coefficient $\hat{\mathbf{c}}_{k/k-1}$ to yield a coefficient estimate $\hat{\mathbf{c}}_{k/k}$. Although other types of digital quantizers exist [1] the quantizer used here (2.2) is the simplest one to implement and yields outstanding results as will be seen in our simulations. The coefficient estimate $\hat{\mathbf{c}}_{k/k}$ is then multiplied by the transformation matrix T to yield the pixel vector estimate $\hat{\mathbf{x}}_{k/k}$. This estimate is then stored in a memory which contains

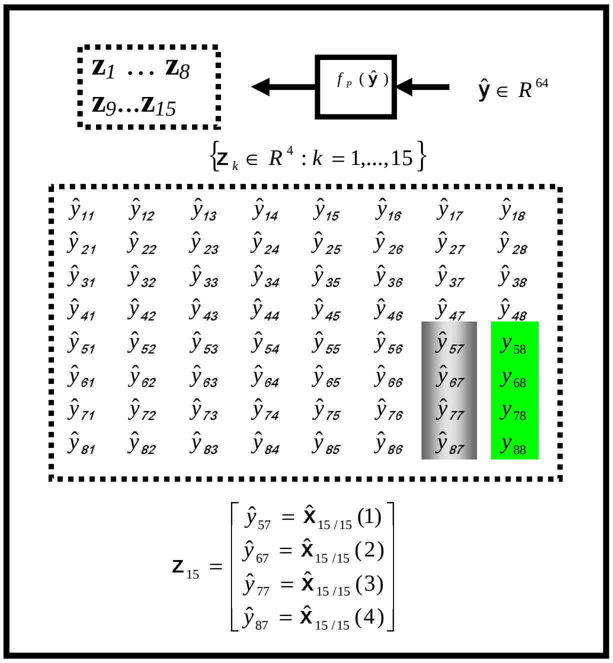

$\mathbf{z}_1 \dots \mathbf{z}_8$
$\mathbf{z}_9 \dots \mathbf{z}_{15}$
$f_P(\hat{\mathbf{y}})$
$\hat{\mathbf{y}} \in R^{64}$
$\{\mathbf{z}_k \in R^4 : k = 1,\dots,15\}$
$\hat{y}_{11}$ $\hat{y}_{12}$ $\hat{y}_{13}$ $\hat{y}_{14}$ $\hat{y}_{15}$ $\hat{y}_{16}$ $\hat{y}_{17}$ $\hat{y}_{18}$
$\hat{y}_{21}$ $\hat{y}_{22}$ $\hat{y}_{23}$ $\hat{y}_{24}$ $\hat{y}_{25}$ $\hat{y}_{26}$ $\hat{y}_{27}$ $\hat{y}_{28}$
$\hat{y}_{31}$ $\hat{y}_{32}$ $\hat{y}_{33}$ $\hat{y}_{34}$ $\hat{y}_{35}$ $\hat{y}_{36}$ $\hat{y}_{37}$ $\hat{y}_{38}$
$\hat{y}_{41}$ $\hat{y}_{42}$ $\hat{y}_{43}$ $\hat{y}_{44}$ $\hat{y}_{45}$ $\hat{y}_{46}$ $\hat{y}_{47}$ $\hat{y}_{48}$
$\hat{y}_{51}$ $\hat{y}_{52}$ $\hat{y}_{53}$ $\hat{y}_{54}$ $\hat{y}_{55}$ $\hat{y}_{56}$ $\hat{y}_{57}$ y_{58}
$\hat{y}_{61}$ $\hat{y}_{62}$ $\hat{y}_{63}$ $\hat{y}_{64}$ $\hat{y}_{65}$ $\hat{y}_{66}$ $\hat{y}_{67}$ y_{68}
$\hat{y}_{71}$ $\hat{y}_{72}$ $\hat{y}_{73}$ $\hat{y}_{74}$ $\hat{y}_{75}$ $\hat{y}_{76}$ $\hat{y}_{77}$ y_{78}
$\hat{y}_{81}$ $\hat{y}_{82}$ $\hat{y}_{83}$ $\hat{y}_{84}$ $\hat{y}_{85}$ $\hat{y}_{86}$ $\hat{y}_{87}$ y_{88}
$\mathbf{z}_{15} = \begin{bmatrix} \hat{y}_{57} = \hat{\mathbf{x}}_{15/15}(1) \\ \hat{y}_{67} = \hat{\mathbf{x}}_{15/15}(2) \\ \hat{y}_{77} = \hat{\mathbf{x}}_{15/15}(3) \\ \hat{y}_{87} = \hat{\mathbf{x}}_{15/15}(4) \end{bmatrix}$

Fig. 4 Image Coding Illustration: Prediction Pre-Processing

the last available estimate $\hat{\mathbf{y}}_{k-1}$ of the pixel matrix **y**. Note that the initial value for $\hat{\mathbf{y}}_{k-1}$, i.e., $\hat{\mathbf{y}}_{0}$, can be any reasonable estimate for each pixel. For instance, since the processing of the image is done in a sequential manner using prediction from pixel block to pixel block, the initial $\hat{\mathbf{y}}_{0}$ can be constructed by assuming for each of its pixel estimates the average value of the pixel block $\mathbf{x}_1$. Fig. 4 shows for the illustrative example how the image estimate at processing stage k=16, i.e., $\hat{\mathbf{y}}_{k-1}= \hat{\mathbf{y}}_{15}$, is used by the predictor pre-processor to generate the pixel estimate predictor pre-processor vector $\mathbf{z}_{15}$. Also note from the same figure how at stage k=16 the 4 scalar elements ($\hat{y}_{57}, \hat{y}_{67}, \hat{y}_{77}, \hat{y}_{87}$) of the 8x8 pixel matrix $\hat{\mathbf{y}}_{15}$ are updated making use of the most recently derived pixel vector estimate $\hat{\mathbf{x}}_{15/15}$. Next the design of the T and P matrices of the PT source coder is reviewed.

The design equations for the T and P matrices are derived by minimizing the mean squared error expression

$$E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k/k})^t(\mathbf{x}_k - \hat{\mathbf{x}}_{k/k})] \tag{2.3}$$

with respect to T and P and subject to three constraints. They are: 1) The elements of $\delta\mathbf{c}_k$ are uncorrelated from each other; 2) The elements of $\delta\mathbf{c}_k$ are zero mean; and 3) The analog quantizer of (2.1) is assumed. After this minimization is performed the following coupled Wiener-Hopf and Eigensystem design equations are derived [5]:

$$P = [I_m \quad 0_{mx1}]\,J\,T\,, \tag{2.4}$$

$$\left\{E[\mathrm{x}_k\mathrm{x}_k^t] - [E[\mathrm{x}_k\mathrm{z}_{k-1}^t] \quad E[\mathrm{x}_k]]\,J\right\}\,T = T\,\Lambda \tag{2.5}$$

$$J = \begin{bmatrix} E[\mathrm{z}_{k-1}\mathrm{z}_{k-1}^t] & E[\mathrm{z}_{k-1}] \\ E[\mathrm{z}_{k-1}^t] & 0 \end{bmatrix}^{-1} \begin{bmatrix} E[\mathrm{z}_{k-1}\mathrm{x}_k^t] \\ E[\mathrm{x}_k^t] \end{bmatrix} \tag{2.6}$$

where these expressions are a function of the first and second order statistics of $\mathbf{x}_k$ and $\mathbf{z}_{k-1}$ including their cross correlation. To find these statistics the following isotropic model for the pixels of **y** can be used [5]:

$$E[y_{ij}] = K\,, \tag{2.7}$$

$$E[(y_{ij} - K)(y_{i+v,j+h} - K) = (P_{avg} - K^2)\rho^D \tag{2.8}$$

$$\rho = E[(y_{ij} - K)(y_{i,j+1} - K)]/(P_{avg} - K^2) \tag{2.9}$$

$$D = \sqrt{(rv)^2 + h^2} \tag{2.10}$$

where v and h are integers, K is the average value of any pixel, P_{avg} is the average power associated with each pixel, and r is a constant that reflects the relative distance between two adjacent vertical and two adjacent horizontal pixels (r =1 when the vertical and horizontal distances are the same).

In Fig. 5 the lossy PT decoder is shown and is noted to be identical in structure to the feedback section of the encoder section of Fig. 2. Next the lossless section of the PT source coder of Fig. 1 is discussed which contains the bit planes methodology advanced in this paper.

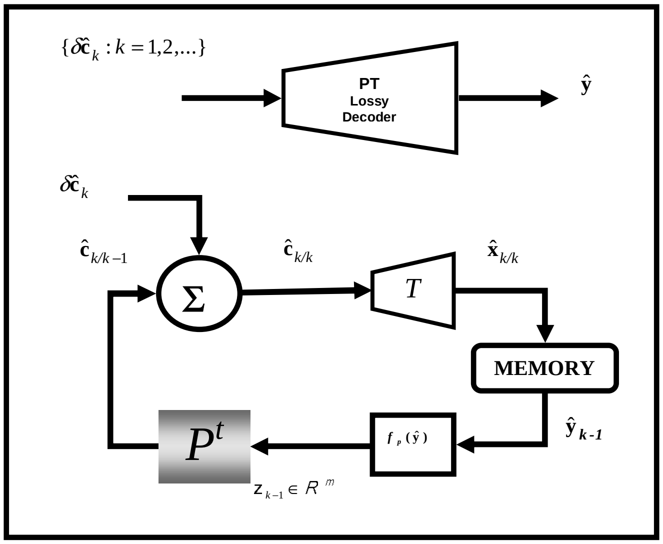


Fig. 5 Lossy PT Decoder

3. BIT PLANES

The general architecture of the offered lossless PT encoder is shown in Fig. 6 which has as input the digitally quantized coefficient error sequence {$\delta\hat{\mathbf{c}}_k$: k=1,..,N_B} where N_B is the total number of coefficient error vectors needed to encode the 2-D image **y**. The output of the lossless PT coder is the desired bit stream $\{b_j \in (0,1) : j = 1,2,.., N_b\}$ where N_b is the number of bits generated by the lossless PT encoder prior to its further encoding using a lossless source coding scheme such as an Arithmetic coder. The coefficient error sequence forms what is called in the figure PT Blocks which is a matrix of dimension n x N_B. In Fig. 7 an illustrative example is presented where n=6 and N_B =6. The most energetic element of each quantized coefficient error is found in the first row of PT Blocks, i.e., in the row {-3 0 0 -1 1 2}, and the least energetic one is found in the last row, i.e., the row {0 0 0 -1 0 0}. PT Blocks is then decomposed into NZ_Amplitude_Locations and NZ_Amplitude_Values. NZ_Amplitude_Locations is an n x N_B dimensional matrix that conveys information about the location of the nonzero (NZ) amplitudes found in PT Blocks. From the simple example of Fig. 7 it is noted that all nonzero elements of PT Blocks are replaced with a 1. NZ_Amplitude_Values, on the other hand, retains the actual NZ amplitude values. In Fig. 7 these amplitudes are shown for our illustrative example where it is noted that the number of elements in each row is not constant and also that no elements are displayed corresponding to the fourth row of PT Blocks since this row is made of zero values only. Returning to Fig. 6 it is noted that the NZ_Amplitude_Locations matrix is now split up into a Boundary matrix and a LocBitPlane block. The Boundary matrix is associated with the location where the zero runs begin in the direction from top to bottom of each column of the NZ_Amplitude_Locations matrix. LocBitPlane, on the other hand, are the bits that remain after the 1's followed by

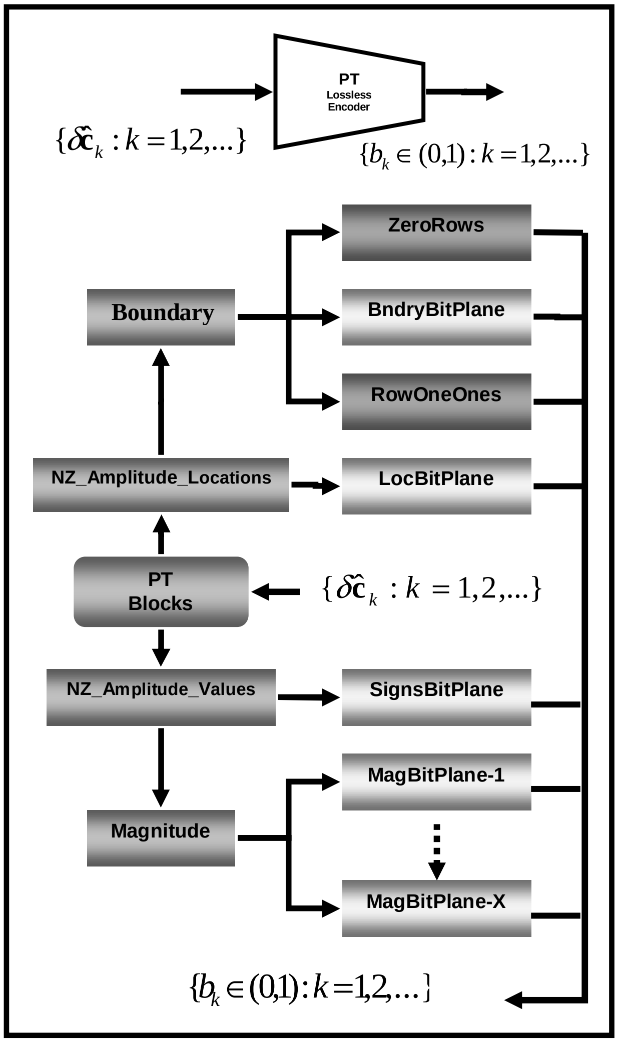
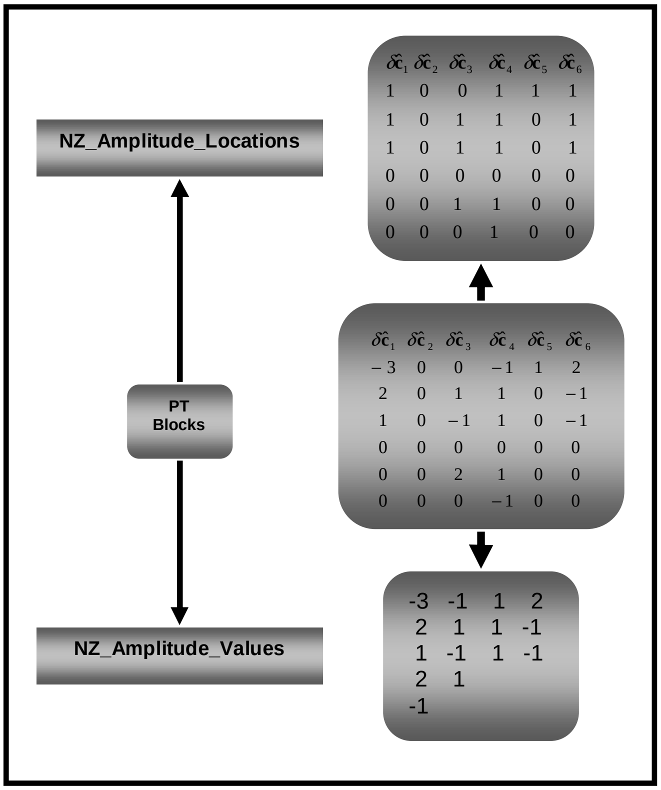


Fig. 6 Lossless PT Encoder



$$\begin{array}{cccccc}
\delta\hat{\mathbf{c}}_1 & \delta\hat{\mathbf{c}}_2 & \delta\hat{\mathbf{c}}_3 & \delta\hat{\mathbf{c}}_4 & \delta\hat{\mathbf{c}}_5 & \delta\hat{\mathbf{c}}_6 \\
1 & 0 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 \\
1 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0
\end{array}$$

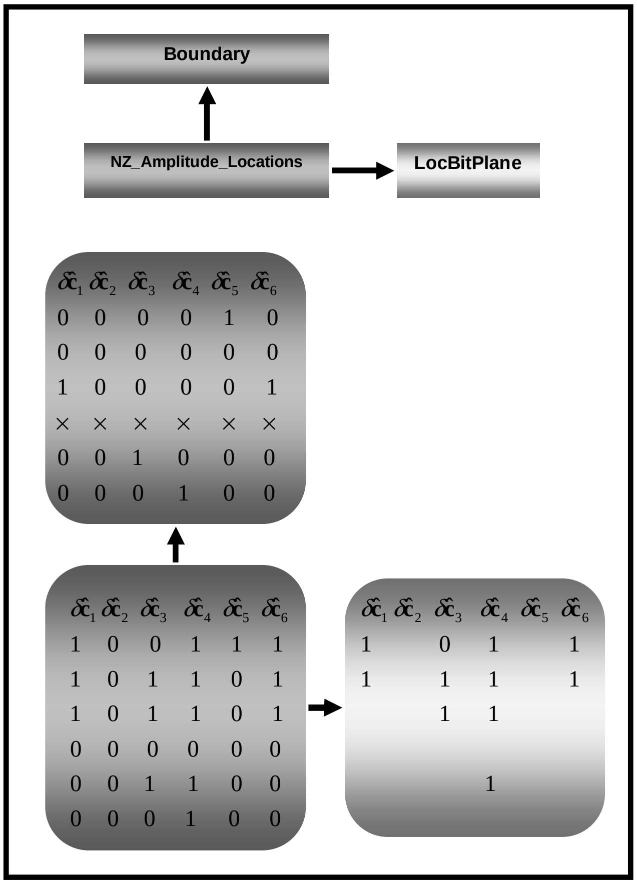
$$\begin{array}{cccccc}
\delta\hat{\mathbf{c}}_1 & \delta\hat{\mathbf{c}}_2 & \delta\hat{\mathbf{c}}_3 & \delta\hat{\mathbf{c}}_4 & \delta\hat{\mathbf{c}}_5 & \delta\hat{\mathbf{c}}_6 \\
-3 & 0 & 0 & -1 & 1 & 2 \\
2 & 0 & 1 & 1 & 0 & -1 \\
1 & 0 & -1 & 1 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2 & 1 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0
\end{array}$$

$$\begin{array}{cccc}
-3 & -1 & 1 & 2 \\
2 & 1 & 1 & -1 \\
1 & -1 & 1 & -1 \\
2 & 1 & & \\
-1 & & &
\end{array}$$

Fig. 7 Illustrative Example: PT Blocks Decomposition

zero runs of the Boundary matrix are eliminated from the NZ_Amplitude_Locations matrix. In Fig. 8 this decomposition is illustrated for the running example. Note that the nonzero Boundary matrix has three symbols. They are 0, 1 and X. The symbol X is used for the elements of a row whose values are all zero, thus it informs us about a zero row. The symbol 1 does not appear more than once for each column and specifies a boundary location where the zero run begins for that particular column. For example, since the zero run starts at row 4 for the first column, the 1 is placed on the third row just prior to the beginning of the zero run. The aforementioned LocBitPlane is also illustrated in Fig. 8. For instance, note how for the third column only the bits {0 1 1} are listed and the zero for the fourth row is ignored since this information is available from the encoding of the Boundary matrix.

Once again returning to Fig. 6 it is now noted that the Boundary matrix is decomposed into three blocks. They are the blocks ZeroRows, BndryBitPlane and RowOneOnes. This decomposition is best explained with the illustrative example of Fig. 9. From this figure it is noted that ZeroRows assigns a 0 to a row of the Boundary matrix if it is composed of the special symbol X, otherwise it assigns a 1 to the row. BndryBitPlane is the same as Boundary matrix except that all rows made up of the special symbol X are removed. In addition BndryBitPlane replaces a 0 with a 1 in the first row of a column with a full zero run. See for example the second column of the Boundary matrix which has a full zero run and for which a 1 has been placed on the first row of the column. Finally RowOneOnes keeps track of the ones in the first row of BndryBitPlane that arose from replacing a 0 with a 1 as mentioned earlier. This completes the encoding of the NZ_Amplitude_Locations matrix of Fig. 6 into bit planes. Next the same is accomplished with the NZ_Amplitude_Values block of Fig. 6 which was illustrated in Fig. 7 with an example.



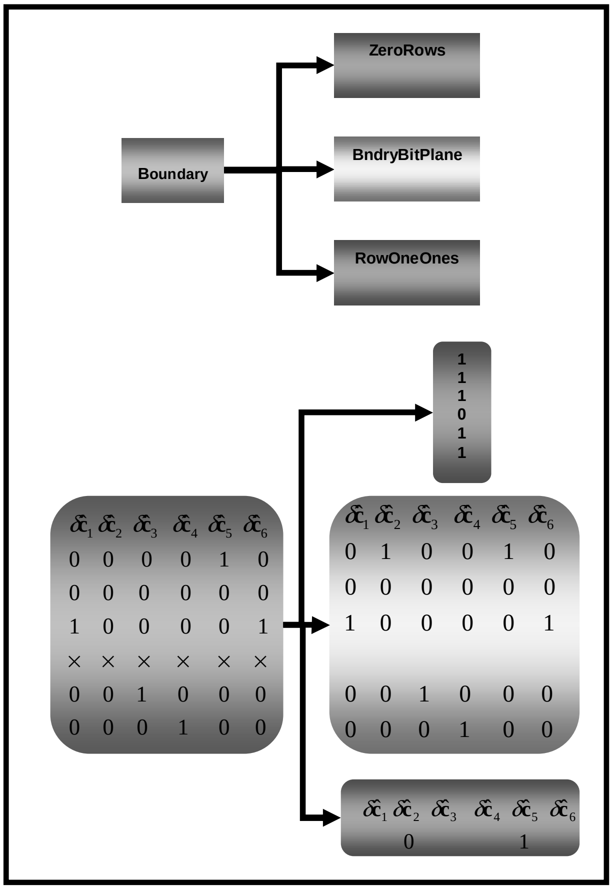
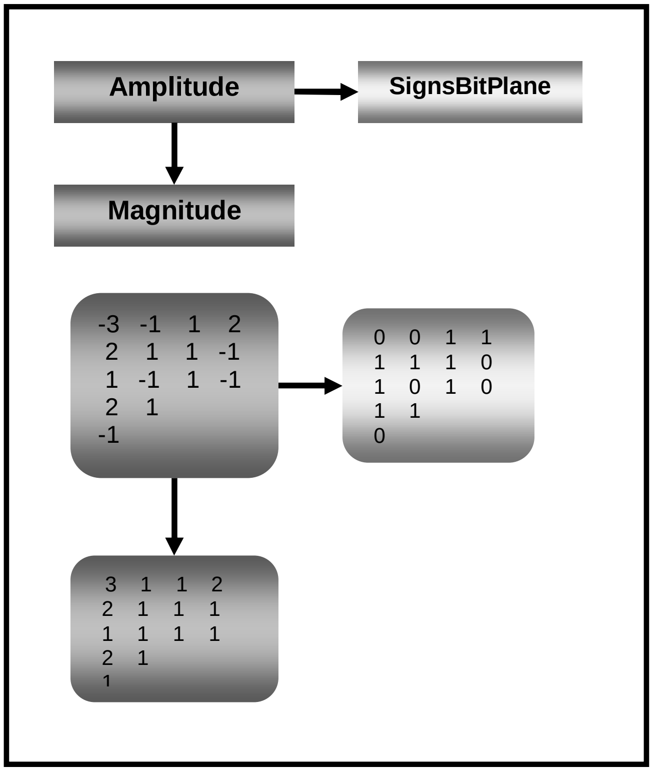


Fig. 8 Illustrative Example: NZ_Amplitude_Locations Decompositions

Fig. 9 Illustrative Example: Boundary Decomposition

From Fig. 6 it is noted that NZ_Amplitude_Values is decomposed into two blocks. One is a Magnitude block and the other is a SignsBitPlane block. The nature of these two blocks is surmised from Fig. 10 which continues our running illustrative example. Note from this figure that the SignsBitPlane block assigns a zero to a negative integer value and a one to a positive integer value. The Magnitude block is self explanatory. Returning for the last time to Fig. 6 it is noted that the Magnitude block is decomposed into X MagBitPlane blocks. Each of these component blocks are readily explained via the illustrative example of Fig. 11. It is first noted that since the maximum integer value for the Magnitude block is 3 there will be 3-1=2 MagBitPlane blocks (it should be noted, however, that if the integer value 2 did not appear in the Magnitude block only one MagBitPlane block is needed with this information sent to the decoder as overhead). MagBitPlane-1 is noted from Fig. 11 to assign a 1 to the integer of magnitude 1 and a 0 to the other cases. On the other hand, MagBitPlane-2 ignores all integers with a magnitude of one, and now assigns a 1 to the integers with a magnitude of 2 and a 0 to the remaining integers. At this point one has the necessary stream of ones and zeros that can then be appropriately encoded using a lossless encoder such as an Arithmetic encoder whose output is then sent to the lossless PT decoder.

In Fig. 12 the lossless PT decoder is shown which receives as input the output of the lossless PT encoder (note that it is assumed here that a lossless decoder such as an Arithmetic decoder was appropriately used to derive this input). The front part of the decoder constructs an n x N_B matrix, ZeroRows_M, made up of either unity rows or zero rows depending on the nature of the ZeroRows bits. In Fig. 13 this construction is illustrated with the running illustrative example. Note that the ZeroRows bits that were derived in Fig. 9 are now used to construct a 6x6 matrix consisting of either unity or zero rows. Next the ZeroRows_M matrix is used in conjunction with the BndryBitPlane bits to generate



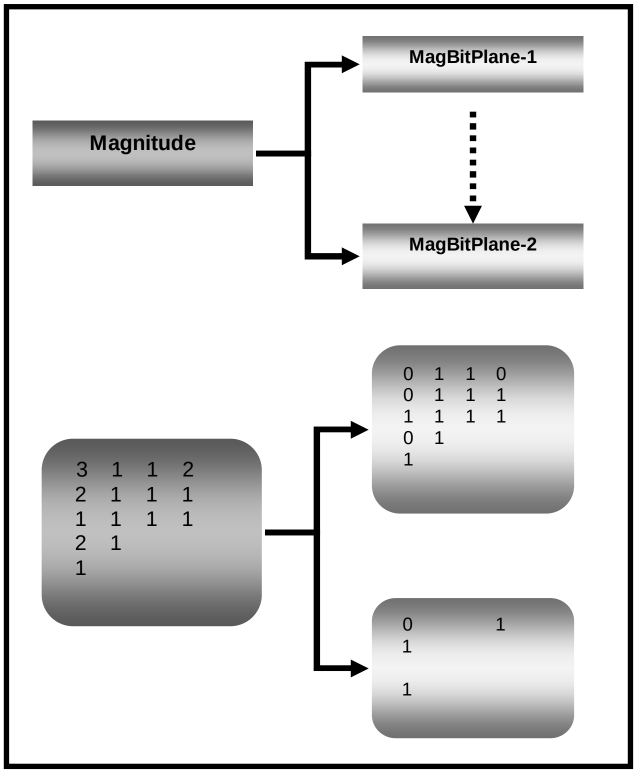
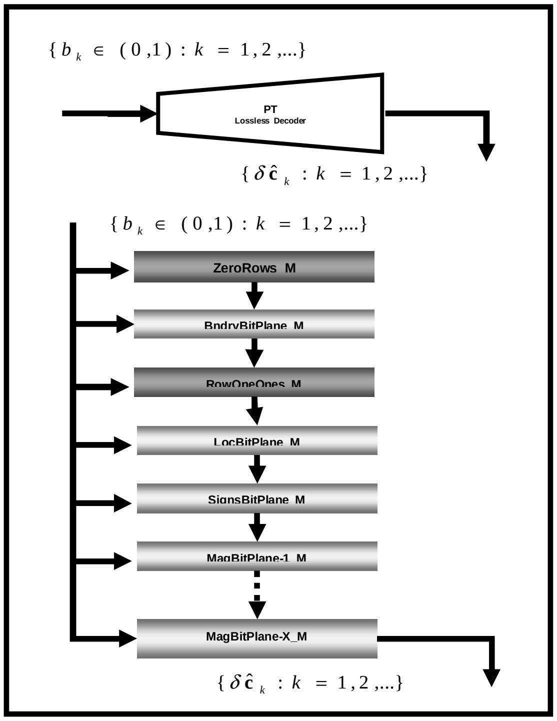


Fig. 10 Illustrative Example: Amplitude Decomposition

Fig. 11 Illustrative Example: Magnitude Decomposition


$\{b_k \in (0,1) : k = 1,2,...\}$
PT
Lossless Decoder
$\{\delta\hat{\mathbf{c}}_k : k = 1,2,...\}$
$\{b_k \in (0,1) : k = 1,2,...\}$
ZeroRows_M
BndryBitPlane_M
RowOneOnes_M
LocBitPlane_M
SignsBitPlane_M
MagBitPlane-1_M
MagBitPlane-X_M
$\{\delta\hat{\mathbf{c}}_k : k = 1,2,...\}$

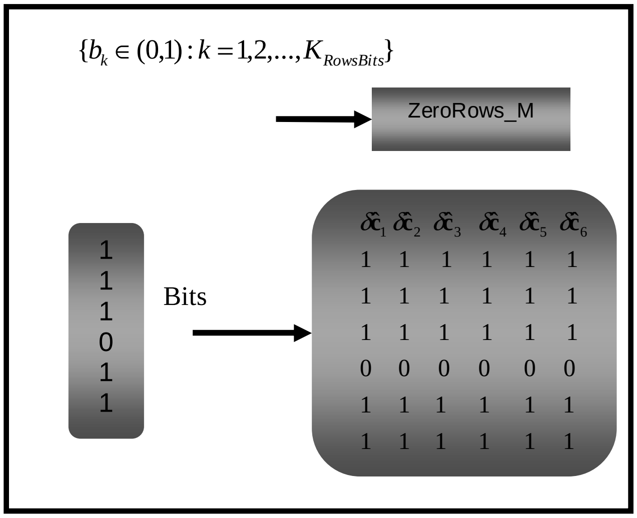

$\{b_k \in (0,1) : k = 1,2,...,K_{RowsBits}\}$
ZeroRows_M
1
1
1
0
1
1
Bits
$\delta\hat{\mathbf{c}}_1$ $\delta\hat{\mathbf{c}}_2$ $\delta\hat{\mathbf{c}}_3$ $\delta\hat{\mathbf{c}}_4$ $\delta\hat{\mathbf{c}}_5$ $\delta\hat{\mathbf{c}}_6$
1 1 1 1 1 1
1 1 1 1 1 1
1 1 1 1 1 1
0 0 0 0 0 0
1 1 1 1 1 1
1 1 1 1 1 1

Fig. 12 Lossless PT Decoder

Fig. 13 Illustrative Example: ZeroRows_M Construction

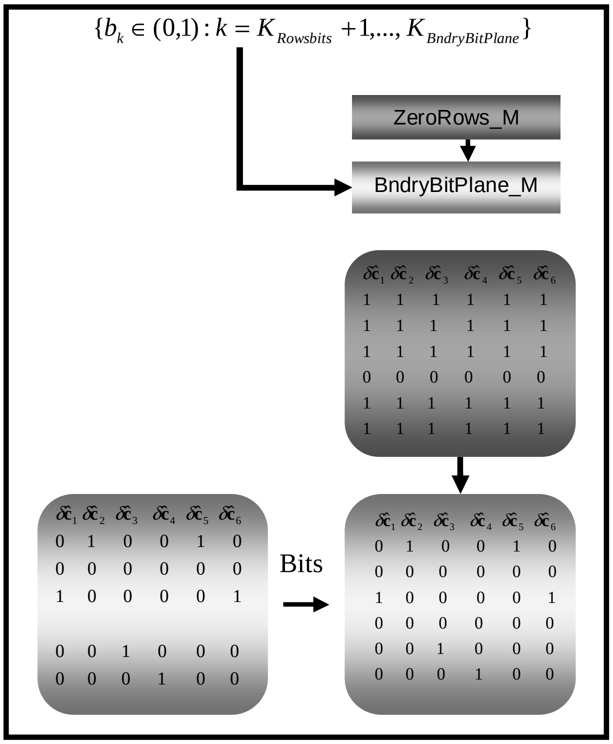

$\{b_k \in (0,1) : k = K_{Rowsbits} + 1,..., K_{BndryBitPlane}\}$
ZeroRows_M
BndryBitPlane_M
$\delta\hat{c}_1$ $\delta\hat{c}_2$ $\delta\hat{c}_3$ $\delta\hat{c}_4$ $\delta\hat{c}_5$ $\delta\hat{c}_6$
1 1 1 1 1 1
1 1 1 1 1 1
1 1 1 1 1 1
0 0 0 0 0 0
1 1 1 1 1 1
1 1 1 1 1 1
$\delta\hat{c}_1$ $\delta\hat{c}_2$ $\delta\hat{c}_3$ $\delta\hat{c}_4$ $\delta\hat{c}_5$ $\delta\hat{c}_6$
0 1 0 0 1 0
0 0 0 0 0 0
1 0 0 0 0 1
0 0 1 0 0 0
0 0 0 1 0 0
Bits
$\delta\hat{c}_1$ $\delta\hat{c}_2$ $\delta\hat{c}_3$ $\delta\hat{c}_4$ $\delta\hat{c}_5$ $\delta\hat{c}_6$
0 1 0 0 1 0
0 0 0 0 0 0
1 0 0 0 0 1
0 0 0 0 0 0
0 0 1 0 0 0
0 0 0 1 0 0

Fig. 14 Illustrative Example: BndrBitPlane_M Construction

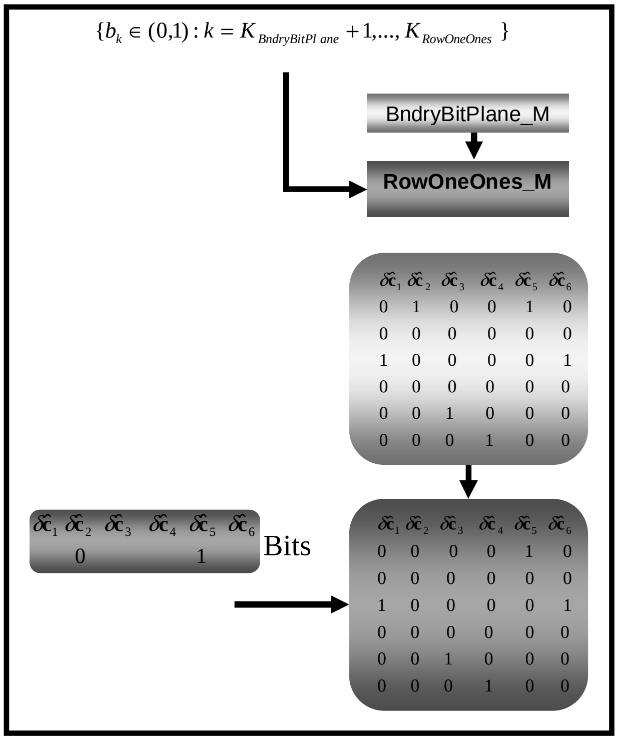

$\{b_k \in (0,1) : k = K_{BndryBitPl\ ane} + 1,..., K_{RowOneOnes}\}$
BndryBitPlane_M
RowOneOnes_M
$\delta\hat{c}_1$ $\delta\hat{c}_2$ $\delta\hat{c}_3$ $\delta\hat{c}_4$ $\delta\hat{c}_5$ $\delta\hat{c}_6$
0 1 0 0 1 0
0 0 0 0 0 0
1 0 0 0 0 1
0 0 0 0 0 0
0 0 1 0 0 0
0 0 0 1 0 0
$\delta\hat{c}_1$ $\delta\hat{c}_2$ $\delta\hat{c}_3$ $\delta\hat{c}_4$ $\delta\hat{c}_5$ $\delta\hat{c}_6$
0 1
Bits
$\delta\hat{c}_1$ $\delta\hat{c}_2$ $\delta\hat{c}_3$ $\delta\hat{c}_4$ $\delta\hat{c}_5$ $\delta\hat{c}_6$
0 0 0 0 1 0
0 0 0 0 0 0
1 0 0 0 0 1
0 0 0 0 0 0
0 0 1 0 0 0
0 0 0 1 0 0

Fig. 15 Illustrative Example: RowOneOnes_M Construction

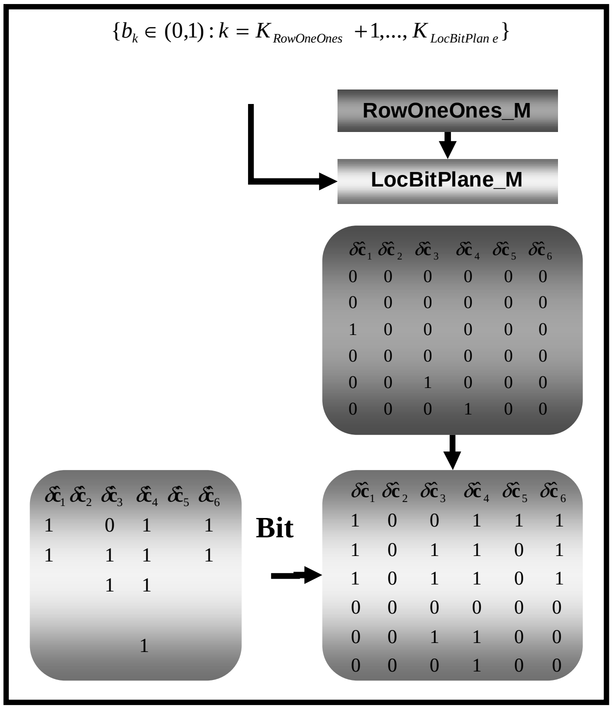

$\{b_k \in (0,1) : k = K_{RowOneOnes} + 1,..., K_{LocBitPlan\ e}\}$
RowOneOnes_M
LocBitPlane_M
$\delta\hat{c}_1$ $\delta\hat{c}_2$ $\delta\hat{c}_3$ $\delta\hat{c}_4$ $\delta\hat{c}_5$ $\delta\hat{c}_6$
0 0 0 0 0 0
0 0 0 0 0 0
1 0 0 0 0 0
0 0 0 0 0 0
0 0 1 0 0 0
0 0 0 1 0 0
$\delta\hat{c}_1$ $\delta\hat{c}_2$ $\delta\hat{c}_3$ $\delta\hat{c}_4$ $\delta\hat{c}_5$ $\delta\hat{c}_6$
1 0 1 1
1 1 1 1
1 1
1
Bit
$\delta\hat{c}_1$ $\delta\hat{c}_2$ $\delta\hat{c}_3$ $\delta\hat{c}_4$ $\delta\hat{c}_5$ $\delta\hat{c}_6$
1 0 0 1 1 1
1 0 1 1 0 1
1 0 1 1 0 1
0 0 0 0 0 0
0 0 1 1 0 0
0 0 0 1 0 0

Fig. 16 Illustrative Example: LocBitPlane_M Construction

the n x N_B matrix BndryBitPlane_M. This process is illustrated in Fig. 14 with the running example. The next step consists of using the derived BndryBitPlane_M matrix together with the RowOneOnes bits to derive a RowOneOnes_M matrix that is also of dimension n x N_B. This process is illustrated once again in Fig. 15 with our running example. Next the RowOneOnes_M matrix is combined with the LocBitPlane bits to derived a LocBitPlane_M matrix of dimension n x N_B. In Fig. 16 this combination is shown for our illustrative example where it is noted that the Loc_Bit_Plane_M matrix is identical to the NZ_Amplitude_Locations matrix shown in Fig. 7. This rather straightforward reconstruction procedure is appropriately continued until the desired error sequence $\{\delta\hat{c}_k$: k=1,..,$N_B\}$ is fully derived. In the next section the proposed algorithm is applied to SAR imagery.

4. A REAL-WORLD APPLICATION

The efficacy of the previously advanced bit planes PT method is now demonstrated by comparing it with wavelets based JPEG2000 in a real-world application. The application consists of compressing 4MB SAR imagery by a factor of 8,192 and then using the decompressed imagery

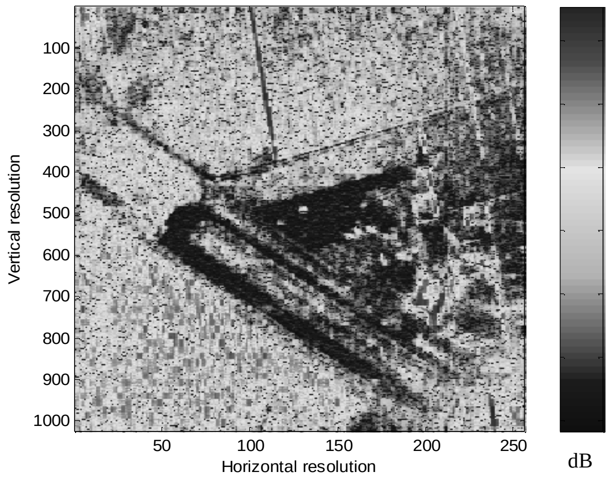


Fig. 17 1024 x 256 4MB SAR Image

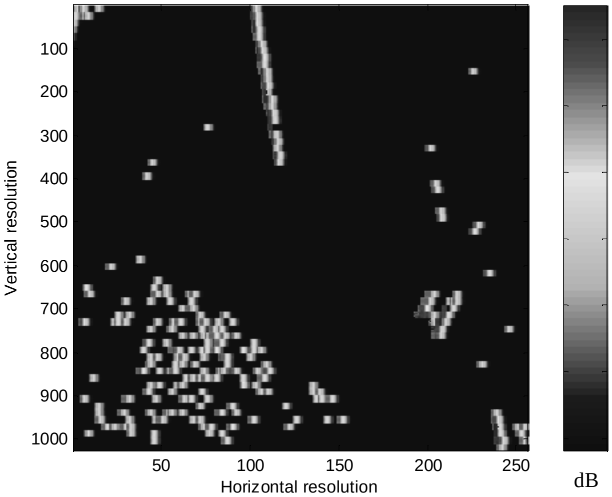


Fig. 18 512 Bytes PT Decompressed SAR Image

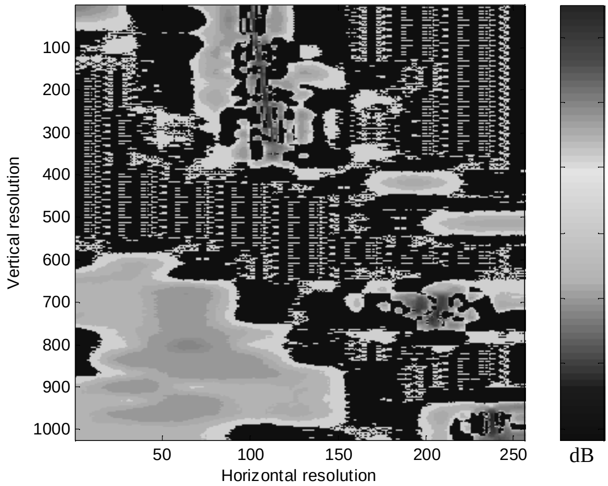


Fig. 19 512 Bytes JPEG2000 Decompressed SAR

as the input to the covariance processor coder of a KA-AMTI radar system subjected to severely taxing environmental disturbances [6]. This SAR imagery is a prior knowledge used in KA-AMTI radar to achieve outstanding SINR radar performance [2].

The 4MB SAR image that will be tested is given in Fig. 17 and is made up of 1024 rows for the down range of 1,500 meters and 256 columns for the cross range of 1,800 meters. The resolution clutter source cell power depicted in this figure is in dBs and is of the Mojave Airport in California. This image was compressed using a 16x1 strip processor that moves on the image from left to right and top to bottom. In Fig. 18 the decompressed SAR image is shown that was derived when the image was compressed by a factor of 8,192 using the PT source coder of this paper. The *SNR*, defined by

$$SNR = 10\log_{10}[\sum_i \sum_j y_{ij}^2 / \sum_i \sum_j (y_{ij} - \hat{y}_{ij})^2], \tag{4.1}$$

performance associated with this image is 12.5 dBs. In Fig. 19 the corresponding decompressed image for JPEG2000 is shown. The SNR performance for this case yields a value of 7.0 dBs which is more than 5 dBs away from the PT approach. Finally, the SINR radar performance derived with JPEG2000 has been found to be at least 2dBs worse than that reported in [6] when using our bit planes PT source coding scheme.

REFERENCES

1. Yeung, R. W., *A First Course in Information Theory*, Kluwer Academic Publishers, New York, 2002
2. Guerci, J.R. and Baranoski, E.J., “Knowledge-Aided Adaptive Radar at DARPA”, *IEEE Signal Processing Magazine*, vol. 23, no. 1, pp. 41-50, January 2006
3. Taubman, D. S. and Marcellin, M., *JPEG2000: Image Compression Fundamentals, Standards and Practice*, Kluwer Academic Publishers, MA, 2002
4. Feria, E.H. and Agaian, S. A, “Accelerated Predictive-Transform”, *Proceedings of IEEE ICASSP*, May 2002
5. Feria, E.H., “Decomposed Predictive-Transform Estimation”, *IEEE Transactions on Signal Processing*, Vol. 42, No. 10, pp. 2811-2822, October 1994
6. —, “Compression-Designs for Knowledge-Aided AMTI Radar Systems”, *Proceedings of SPIE Defense and Security Symp.*, April 2006

ACKNOWLEDGEMENTS

Dr. Joseph R. Guerci, the Director of DARPA’s Special Projects Office (SPO), and Dr. Edward Baranoski, the Program Manager for DARPA’s Knowledge-Aided Sensory Signal Processing Expert Reasoning (KASSPER), are gratefully acknowledged for their guidance during the course of this research. Their objective criticisms, advice and suggestions at every stage of this project are highly appreciated.