A New Approach to Compatible Adaptive Block-size Transforms

DONG Jie

Institute of Information and Communication Engineering

Zhejiang University

Outline

- Background
  - ICT-based ABT
  - Basic Theory of ICT
- Problem Formulation and Objective
- Compatible ABT Algorithm
- An Example of Compatible ABT
- Complexity Analysis
- Conclusion

ICT-based ABT

- ABT: Adaptive Block-size Transform
  - Choosing the transform size adaptively according to the signal properties
  - Making a better trade off between the bit rate and the visual quality
- ICT: Integer Cosine Transform
  - The coefficients in ICT matrix are simple integers
    - Almost the same compression efficiency as DCT
    - Simple integer arithmetic implementation
    - Without mismatch among decoders
- ICT-based ABT: Using ICT as the transform method in an ABT coding system.

Basic Theory of ICT

Stage 1. Integer Transform

$$\mathrm{F}_{n\times m} = \mathrm{ICT}_n \times \mathrm{f}_{n\times m} \times \mathrm{ICT}_m^{\mathrm{T}}$$

$\mathrm{f}_{n\times m}$: nxm input data

$\mathrm{F}_{n\times m}$: nxm output of Integer Transform

ICT_n : nxn ICT kernel for vertical transform

ICT_m : mxm ICT kernel for horizontal transform

$\times$: matrix multiplication operator

Basic Theory of ICT

Stage 2. Scaling Process $\mathrm{F}_{n\times m} // \mathrm{R}_{n\times m}$

$\mathrm{R}_{n\times m}$: intermediate scaling matrix

$//$: element division operator

$\mathrm{R}_{n\times m}$ can be obtained by $\mathrm{R}_{n\times m} = \mathrm{R}_{n\times 1} \times \mathrm{R}^{\mathrm{T}}_{m\times 1}$

$\mathrm{R}_{n\times 1}$: n×1 matrix

$\mathrm{R}_{n\times 1}(i)$: the norm of the i^{th} row vector in ICT_n

In practical implementation, scaling process is $\left(\mathrm{F}_{n\times m} \otimes \mathrm{P}_{n\times m}\right) >> N$

$\mathrm{P}_{n\times m}$: scaling matrix

$\otimes$: element multiplication operator

$>>$: right shifting operator

Problem Formulation and Objective

Problem Formulation

- For variable block-size ICT
  - Separate integer transform units
  - Different scaling matrices

  ICT-based ABT consumes a vast amount of resources in practical implementations

Our objective

- Reducing the implemented complexity of an ICT-based ABT system with block-size 8x8, 8x4, 4x8 and 4x4
  - The compatibility of integer transform units
  - The compatibility of scaling matrix so as to reduce memory requirement
  - Reducing computational burden
  - Maintaining coding efficiency

Compatible ABT Algorithm

- The compatibility of integer transform units
  - Choosing compatible ICT kernels
  - The way to reuse the butterfly structure
- The compatibility of scaling matrix
  - Analysis of the relationship among different block-size scaling matrices
  - Pre-scaled Integer Transform (PIT)
  - The way to use the compatible scaling matrix

Compatible Integer Transform

- The 4 block-size integer transforms

$$F_{4\times4} = ICT_4 \times f_{4\times4} \times ICT_4^T \qquad F_{4\times8} = ICT_4 \times f_{4\times8} \times ICT_8^T$$

$$F_{8\times4} = ICT_8 \times f_{8\times4} \times ICT_4^T \qquad F_{8\times8} = ICT_8 \times f_{8\times8} \times ICT_8^T$$

- Two butterfly structures needed
  - 8-point
  - 4-point

Compatible Integer Transform

- The ICT kernel ICT_8 and ICT_4 are denoted as

$$ICT_8 = \begin{bmatrix} n_0 & n_0 & n_0 & n_0 & n_0 & n_0 & n_0 & n_0 \\ n_1 & n_3 & n_5 & n_7 & -n_7 & -n_5 & -n_3 & -n_1 \\ n_2 & n_6 & -n_6 & -n_2 & -n_2 & -n_6 & n_6 & n_2 \\ n_3 & -n_7 & -n_1 & -n_5 & n_5 & n_1 & n_7 & -n_3 \\ n_4 & -n_4 & -n_4 & n_4 & n_4 & -n_4 & -n_4 & n_4 \\ n_5 & -n_1 & n_7 & n_3 & -n_3 & -n_7 & n_1 & -n_5 \\ n_6 & -n_2 & n_2 & -n_6 & -n_6 & n_2 & -n_2 & n_6 \\ n_7 & -n_5 & n_3 & -n_1 & n_1 & -n_3 & n_5 & -n_7 \end{bmatrix}$$

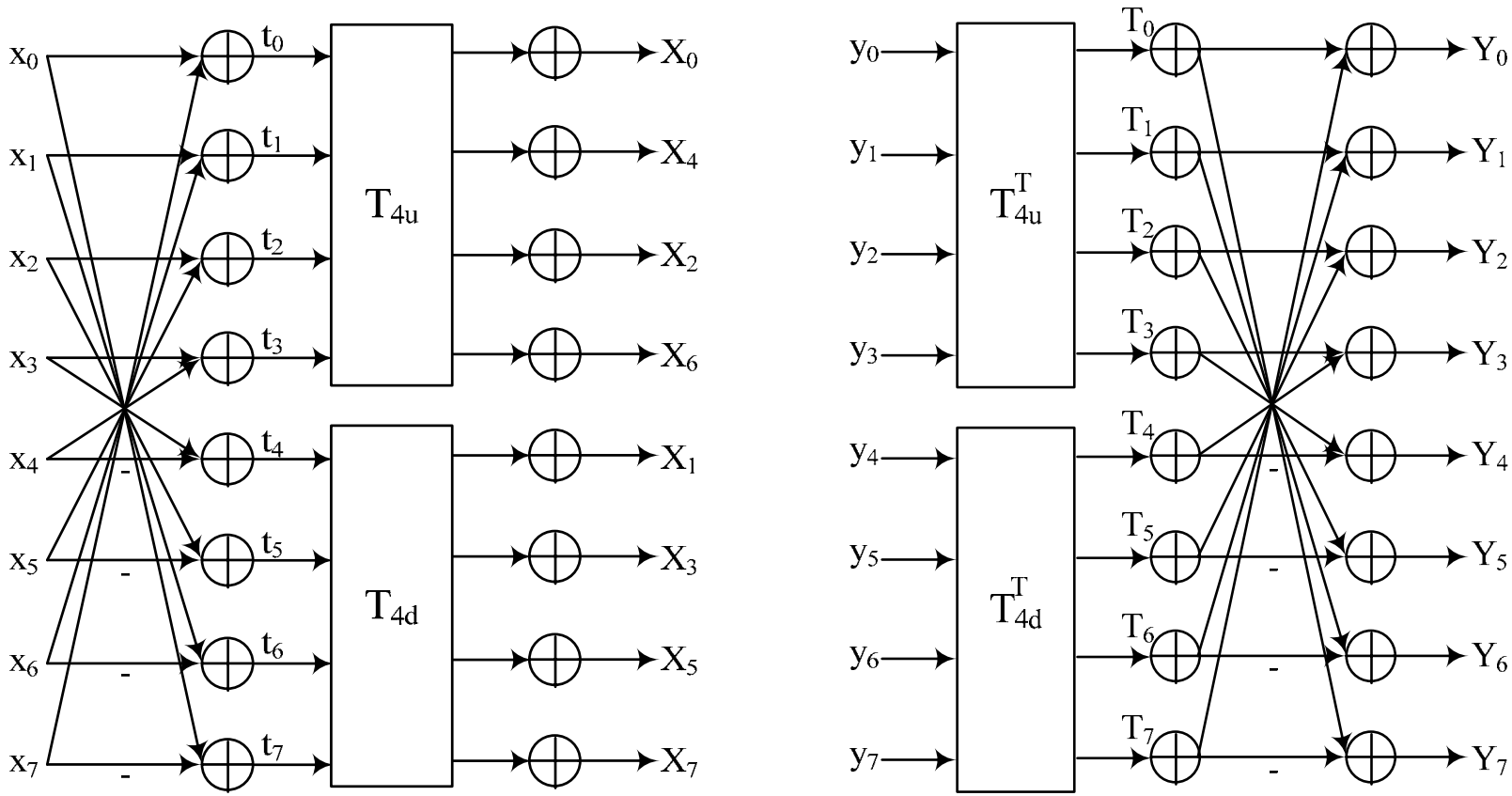
$$ICT_4 = \begin{bmatrix} m_0 & m_0 & m_0 & m_0 \\ m_1 & m_3 & -m_3 & -m_1 \\ m_2 & -m_2 & -m_2 & m_2 \\ m_3 & -m_1 & m_1 & -m_3 \end{bmatrix}$$

- T_{4u} and T_{4d} can always be extracted from ICT_8

$$T_{4u} = \begin{bmatrix} n_0 & n_0 & n_0 & n_0 \\ n_2 & n_6 & -n_6 & -n_2 \\ n_4 & -n_4 & -n_4 & n_4 \\ n_6 & -n_2 & n_2 & -n_6 \end{bmatrix} \qquad T_{4d} = \begin{bmatrix} n_1 & n_3 & n_5 & n_7 \\ n_3 & n_7 & -n_1 & -n_5 \\ n_5 & -n_1 & -n_7 & n_3 \\ n_7 & -n_5 & n_3 & -n_1 \end{bmatrix}$$

- Choose T_{4u} as ICT_4

Compatible Butterfly Structure



The 8-point forward and inverse transform butterfly structures

Compatible Scaling Matrix

Relationship between P_{nxm}

- Precondition: T_{4u} is chosen as ICT_4

$$P_{4\times4}(i,j) = P_{8\times8}(2i,2j)\times 2$$

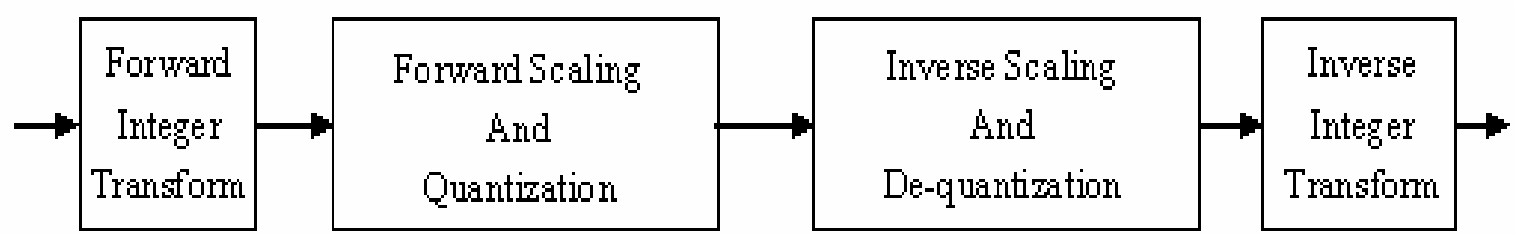
$$P_{8\times4}(i,j) \approx P_{8\times8}(i,2j)\times\sqrt{2}$$

$$P_{4\times8}(i,j) \approx P_{8\times8}(2i,j)\times\sqrt{2}$$

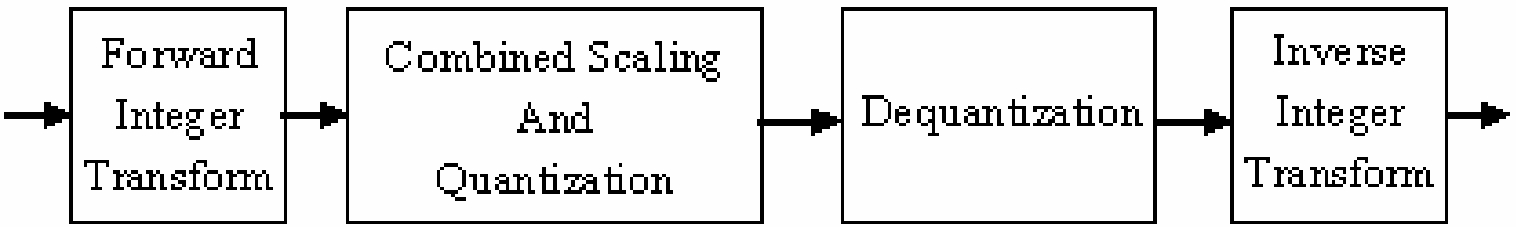
The magnitude of the elements from the i^{th} row and the j^{th} column of $P_{4\times4}$, $P_{4\times4}(i, j)$, are nicely twice as large as that from the $2i^{th}$ row and the $2j^{th}$ column of $P_{8\times8}$, $P_{8\times8}(2i, 2j)$.

Pre-scaled Integer Transform

- The scaling process of the inverse ICT is moved to the encoder side and combined with that of the forward ICT as one single process.



Block diagram of a conventional ICT scheme



Block diagram of a PIT scheme

Compatible Scaling Matrix

Relationship between P_{nxm}

- Preconditions
  - T_{4u} is chosen as ICT_4
  - PIT is introduced

$$P_{4\times4}(i,j) = P_{8\times8}(2i,2j) << 2$$
$$P_{8\times4}(i,j) = P_{8\times8}(i,2j) << 1$$
$$P_{4\times8}(i,j) = P_{8\times8}(2i,j) << 1$$

All P_{4x4}, P_{4x8}, P_{8x4} are involved in P_{8x8}, despite different shifting operations, thus achieving the compatibility of scaling matrix.

An Example of Compatible ABT

The ICT Kernel Selection

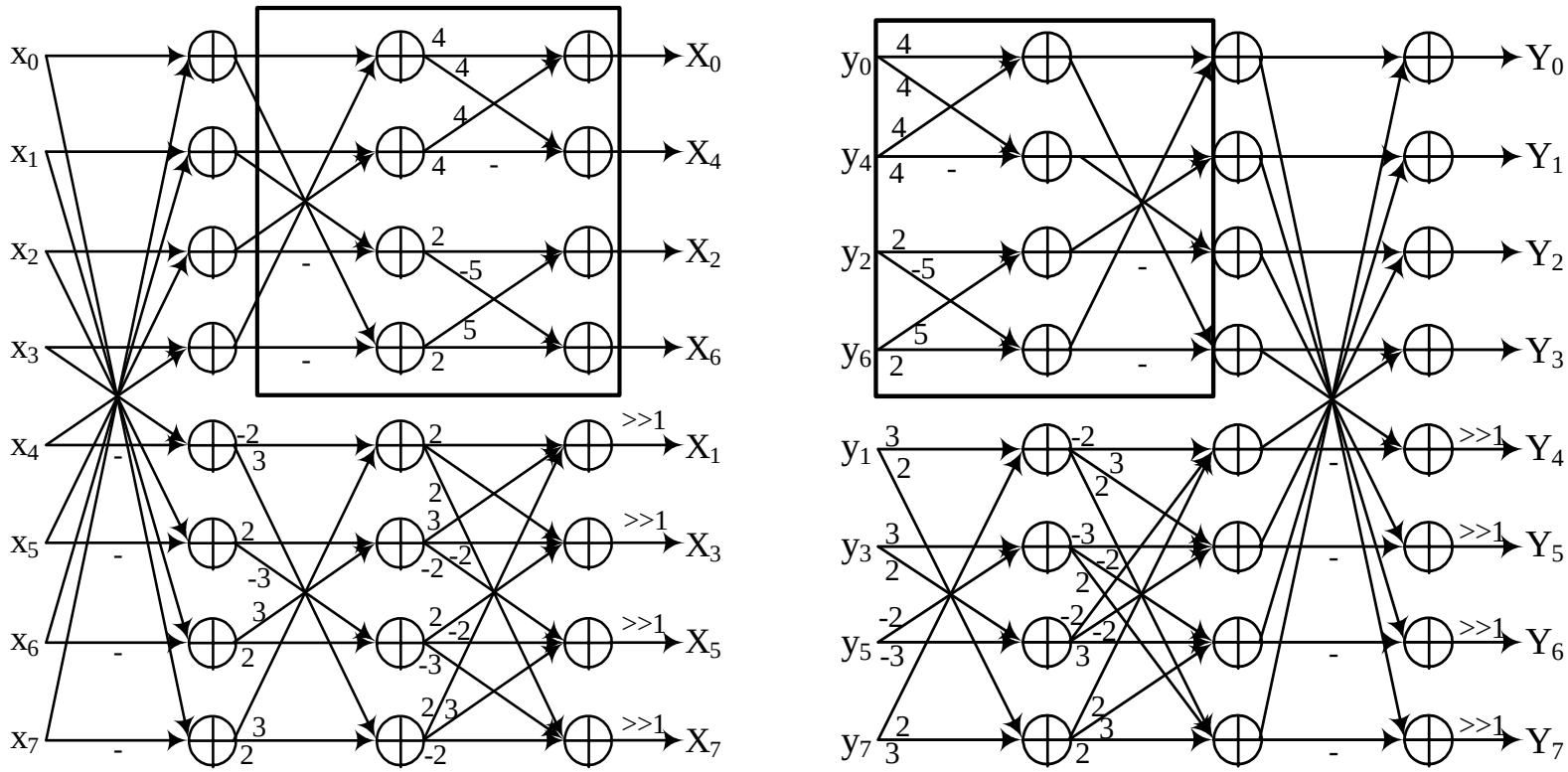
$$ICT_8 = \begin{bmatrix} 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 10 & 9 & 6 & 2 & -2 & -6 & -9 & -10 \\ 10 & 4 & -4 & -10 & -10 & -4 & 4 & 10 \\ 9 & -2 & -10 & -6 & 6 & 10 & 2 & -9 \\ 8 & -8 & -8 & 8 & 8 & -8 & -8 & 8 \\ 6 & -10 & 2 & 9 & -9 & -2 & 10 & -6 \\ 4 & -10 & 10 & -4 & -4 & 10 & -10 & 4 \\ 2 & -6 & 9 & -10 & 10 & -9 & 6 & -2 \end{bmatrix} / 2$$

$$T_{4u} = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 5 & 2 & -2 & -5 \\ 4 & -4 & -4 & 4 \\ 2 & -5 & 5 & -2 \end{bmatrix}$$

$$T_{4d} = \begin{bmatrix} 10 & 9 & 6 & 2 \\ 9 & 2 & -10 & -6 \\ 6 & -10 & -2 & 9 \\ 2 & -6 & 9 & -10 \end{bmatrix} / 2$$

$$ICT_4 = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 5 & 2 & -2 & -5 \\ 4 & -4 & -4 & 4 \\ 2 & -5 & 5 & -2 \end{bmatrix}$$

Processing 4-point Transform in 8-point Transform Butterfly Structure



The forward and inverse transform butterfly structures for ICT_8

Scaling Matrices for the Encoders

$$P_{8\times8}=\begin{bmatrix}8192 & 9489 & 9039 & 9489 & 8192 & 9489 & 9039 & 9489\\ 9489 & 10992 & 10471 & 10992 & 9489 & 10992 & 10471 & 10992\\ 9039 & 10471 & 9975 & 10471 & 9039 & 10471 & 9975 & 10471\\ 9489 & 10992 & 10471 & 10992 & 9489 & 10992 & 10471 & 10992\\ 8192 & 9489 & 9039 & 9489 & 8192 & 9489 & 9039 & 9489\\ 9489 & 10992 & 10471 & 10992 & 9489 & 10992 & 10471 & 10992\\ 9039 & 10471 & 9975 & 10471 & 9039 & 10471 & 9975 & 10471\\ 9489 & 10992 & 10471 & 10992 & 9489 & 10992 & 10471 & 10992\end{bmatrix}$$

$$P_{8\times4}=\begin{bmatrix}8192 & 9039 & 8192 & 9039\\ 9489 & 10471 & 9489 & 10471\\ 9039 & 9975 & 9039 & 9975\\ 9489 & 10471 & 9489 & 10471\\ 8192 & 9039 & 8192 & 9039\\ 9489 & 10471 & 9489 & 10471\\ 9039 & 9975 & 9039 & 9975\\ 9489 & 10471 & 9489 & 10471\end{bmatrix}$$

$$P_{4\times4}=\begin{bmatrix}8192 & 9039 & 8192 & 9039\\ 9039 & 9975 & 9039 & 9975\\ 8192 & 9039 & 8192 & 9039\\ 9039 & 9975 & 9039 & 9975\end{bmatrix}$$

$$P_{4\times8}=\begin{bmatrix}8192 & 9489 & 9039 & 9489 & 8192 & 9489 & 9039 & 9489\\ 9039 & 10471 & 9975 & 10471 & 9039 & 10471 & 9975 & 10471\\ 8192 & 9489 & 9039 & 9489 & 8192 & 9489 & 9039 & 9489\\ 9039 & 10471 & 9975 & 10471 & 9039 & 10471 & 9975 & 10471\end{bmatrix}$$

Simulation and Test Condition

- Simple 16-bit integer arithmetic
- Integrated into JM50c

Stream Structure	IPPP… and IBBPBBP…
Frame Rate	60 fps
Intra Period	0.5 s
Search Range	± 32
Inter-frame Search	All modes Enabled
Reference Frame	2
Entropy Coding Method	CABAC
QP	Fixed (20, 24, 28, 32)
RDO	On

Experimental Results

Sequence	Gain (dB) with stream structure IPPPPPP	Gain (dB) with stream structure IBBPBBP
City	0.001	0.003
Harbour	0.050	0.006
Night	0.006	0.001
Raven	0.005	0.001

The performance of the proposed example is a little better than that of JM50c.

Complexity Analysis

- The area for an independent 4-point butterfly structure can be saved.
- Decoder
  - No scaling process, thus reducing computational burden.
  - No scaling matrix, thus saving 144bytes.
  - If scaling process is combined with dequantization, 144xK bytes will be saved for totally K different QP.
- Encoder
  - A single 8x8 scaling matrix stored, thus saving 80xK 16-bit integers for totally K different QP if scaling process is combined with quantization.

Note—K is equal to 6 in H.264 due to QP period.

Conclusion

- An approach to compatible ICT-based ABT is presented.
  - Compatibility of integer transform
  - Compatibility of scaling matrix
- Greatly saving hardware resources
  - Only 8-point butterfly structure needed
  - Single scaling matrix for encoder
  - No scaling matrix stored for the decoder
  - Reducing the computational burden of decoder
- Without loss of coding efficiency

Thank you!