High-Accuracy Implementation of Fast DCT Algorithms Based on Algebraic Integer Encoding

Maxim Vashkevich[1], Marek Parfieniuk[2], Alexander Petrovsky[1,2]

[1] Computer Engineering Department, Belarusian State University of Informatics and Radioelectronics
Minsk, Belarus

[2] Department of Digital Media and Computer Graphics, Bialystok University of Technology
Bialystok, Poland

Int. Conf. on Signals and Electronic Systems (ICSES) 2012
18–21 September 2012, Wroclaw, Poland

The motivation

The need for new algorithms and circuits for approximating the DCT

- future video/image coding standards are developed, like HEVC (High-Efficiency Video Coding)
- new standards use transforms of sizes greater than 8, which are not supported by the existing infrastructure
- new design/implementation approaches have appeared that need verification, as they possibly allow improved transforms and circuits to be developed

Our goal is to develop a method for synthesizing and implementing DCT

- that uses the Algebraic Signal Processing (ASP) to minimize number of operations and to obtain regular signal flow graphs
- that uses the Algebraic Integer Encoding (AIE) to enable multiplierless implementation and high accuracy

Algebraic signal processing

A general theory by Pueschel[1]

- connects discrete transforms with algebraic structures
- allows results of algebra to be used to optimize computations

In ASP, a transform is associated with a polynomial algebra:

$$\mathcal{A}_{\mathbb{F}} = \mathbb{F}[x]/p(x) \tag{1}$$

where

- $\mathbb{F}$ is the set of all polynomials in x with coefficients of some number field.
- polynomials of $\mathcal{A}_{\mathbb{F}}$
  - have degrees smaller than $\deg(p)$
  - are added and multiplied modulo $p(x)$

[1] M. Püschel and J. Moura, "Algebraic signal processing theory: Cooley-Tukey type algorithms for DCTs and DSTs," *IEEE Trans. Signal Process.*, vol. 56, no. 4, pp. 1502–1521, Apr. 2008

Algebra as a model of a polynomial transform

For a fixed basis

$$b = [p_0(x), \ldots, p_{n-1}(x)]$$

the algebra determines the polynomial transform that is represented by the matrix

$$P_{b,\alpha} = [p_l(\alpha_k)]_{0 \le k,l < n}$$

where α_k are roots of $p(x)$.

Factorizations of the polynomials and decompositions of the algebra can be used to model decompositions of the polynomial transform.

If the transform matrix is decomposed into sparse matrices, a fast algorithm is developed.

ASP-based derivation of fast transforms

The factorization

$$p(x) = q(x)r(x)$$

corresponds to the algebra decomposition:

$$\begin{aligned} \mathbb{F}[x]/p(x) &\rightarrow \mathbb{F}[x]/q(x) \oplus \mathbb{F}[x]/r(x) & (2)\\ &\rightarrow \bigoplus_{0\le i<k} \mathbb{F}[x]/(x-\beta_i) \oplus \bigoplus_{0\le j<m} \mathbb{F}[x]/(x-\gamma_j) & (3)\\ &\rightarrow \bigoplus_{0\le i<n} \mathbb{F}[x]/(x-\alpha_i) & (4) \end{aligned}$$

and to the factorization of the transform matrix:

$$\mathcal{P}_{b,\alpha} = P(\mathcal{P}_{c,\beta} \oplus \mathcal{P}_{d,\gamma})B, \qquad (5)$$

where

- P is a permutation matrix
- B is a sparse matrix with ± 1 as non-zero elements
- $\mathcal{P}_{c,\beta}$ and $\mathcal{P}_{d,\gamma}$ are half-size matrices (simpler transforms)

The type 2 DCT (DCT-2) can be modeled as

$$\mathbb{Q}[x]/(x-1)U_{n-1}(x), \quad b = (V_0(x), \ldots, V_{n-1}(x)) \tag{6}$$

The type 4 DCT (DCT-4) can be modeled as

$$\mathbb{Q}[x]/2T_n(x), \quad b = (V_0(x), \ldots, V_{n-1}(x)) \tag{7}$$

where

- $\mathbb{Q}$ denotes the field of rational numbers
- $T_n(x)$, $U_n(x)$ and $V_n(x)$ denote Chebyshev polynomials of 1st, 2nd, and 3rd kind, respectively

For Chebyshev polynomials, a lot of properties and factorizations are known, which can be used to develop fast transforms.

Fast recursive algorithm for DCT-2 of a power-of-two size ($n = 2^k$)

For the DCT-2, the following factorizations of the Chebyshev polynomial of second kind:

$$U_{2n-1}(x) = U_{n-1}(x) \cdot 2T_n(x), \tag{8}$$

leads to the transform decomposition

$$\overline{\text{DCT-2}}_{2n} = L_n^{2n}(\overline{\text{DCT-2}}_n \oplus \overline{\text{DCT-4}}_n)B_{2n}, \tag{9}$$

into half-size transforms combined via trivial additions/permutations.

One of the subtransforms is the type 4 DCT, which needs another fast algorithm.

Fast recursive algorithm for DCT-4 of a power-of-two size ($n = 2^k$)

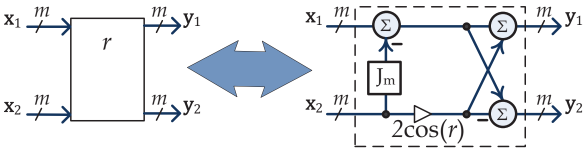
In order to obtain a fast DCT-4 algorithm, we propose to use the factorization

$$2T_{2n}(x) \; - \; 2\cos r\pi \quad = \quad \left(2T_n(x) - 2\cos\tfrac{r\pi}{2}\right) \; \times \; \left(2T_n(x) - 2\cos\pi(1-\tfrac{r}{2})\right) \qquad (10)$$

which lead to the recursive formula

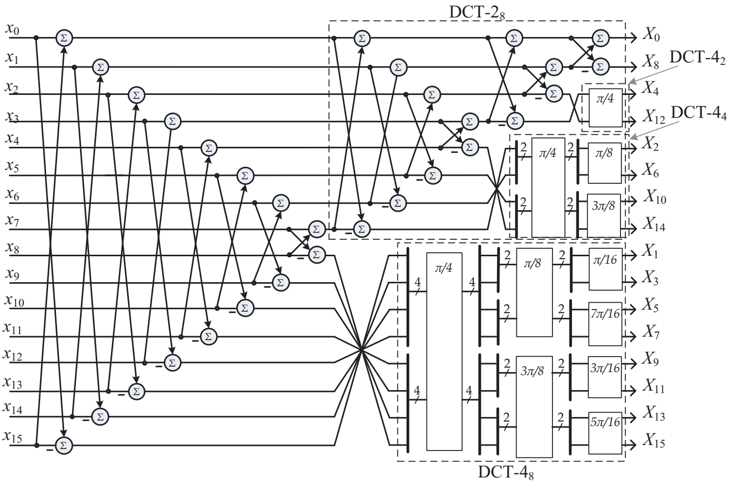
$$\overline{\text{DCT-4}}_{2n}(r) = P \cdot (\overline{\text{DCT-4}}_n(\tfrac{r}{2}) \oplus \overline{\text{DCT-4}}_n(1-\tfrac{r}{2})) \cdot B_{2n}^{(C4)}(r) \qquad (11)$$

where only $B_{2n}^{(C4)}(r)$ is the key block which cannot be recursively decomposed and is related to nontrivial multiplications.



Signal flow graph of the developed fast algorithm for 16-point DCT-2

The joint use of the presented results leads to the following graph



It must be supported by efficient implementation of the key blocks.

The idea of algebraic integer encoding

Algebraic integers (AIs) are defined by roots of monic[2] polynomials with integer coefficients.

For example,

$$z = 2\cos(\pi/8) = \sqrt{2+\sqrt{2}}$$

is the root of the polynomial

$$p(z) = z^4 - 4z^2 + 2$$

Based on such a root, algebraic integers are defined using the polynomial expansion

$$f(z) = a_0 z^0 + a_1 z^1 + \cdots + a_{n-1} z^{n-1},$$

where a_i are integers

For a carefully selected root, even an irrational value can be encoded as a vector of integer values of the expansion coefficients.

[2] The leading coefficient is 1.

AI encoding of DCT multipliers

Since all DCT multipliers have the form

$$2cos\frac{k\pi}{2n}$$

it is reasonable to define

$$z = 2cos\frac{k\pi}{2n}$$

In this case, the polynomial $p(z)$ and all multipliers can be expressed using the Chebyshev polynomials of the first kind:

$$p(z) = 2T_n(z/2),$$

and

$$2cos\frac{k\pi}{2n} = 2T_k(z/2).$$

AI encoding of DCT multipliers (cont.)

For $\mathrm{DCT}\text{-}4_2$ ($z = 2\cos(\pi/4)$)

$p(z) = z^2 - 2$

$2\cos(\pi/4) = z$

For $\mathrm{DCT}\text{-}4_4$ ($z = 2\cos(\pi/8)$)

$p(z) = z^4 - 4z^2 + 2$

$2\cos(\pi/8) = z$

$2\cos(2\pi/8) = z^2 - 2$

$2\cos(3\pi/8) = z^3 - 3z$

For $\mathrm{DCT}\text{-}4_8$ ($z = 2\cos(\pi/16)$)

$p(z) = z^8 - 8z^6 + 20z^4 - 16z^2 + 2$

$2\cos(\pi/16) = z$

$2\cos(2\pi/16) = z^2 - 2$

$2\cos(3\pi/16) = z^3 - 3z$

$2\cos(4\pi/16) = z^4 - 4z^2 + 2$

$2\cos(5\pi/16) = z^5 - 5z^3 + 5z$

$2\cos(6\pi/16) = z^6 - 6z^4 + 9z^2 - 2$

$2\cos(7\pi/16) = z^7 - 7z^5 + 14z^3 - 7z$

Computations based on the AIE

- Instead of multiplying input values by coefficients, the values are multiplied by AI codes so as to give scaled codes
- As AI codes are integers, the multiplications can be realized with bit shifts and additions.
- At the output, values of transform coefficients can recovered by substituting the scaled codes and z into the expansion $f(z)$

Final reconstruction step (FRS) for AI decoding

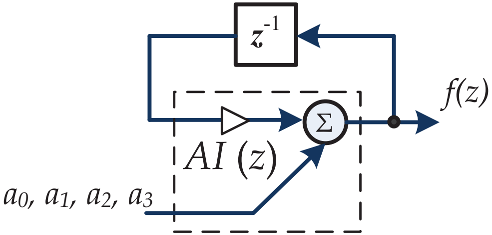
Instead of using the direct expansion

$$f(z) = a_0 z^0 + a_1 z^1 + \cdots + a_{n-1} z^{n-1}$$

it is more efficient to use Horner's rule:

$$f(z) = (\ldots(a_{n-1}z + a_{n-2})z + a_{n-3})z + \cdots + a_1)z + a_0.$$

which is realized by the following FRS circuit



Obtainable accuracy of the developed approximations of the DCT-16

The accuracy measured in terms of coding gain: a measure of how well a transform compacts the energy of signal into a small number of coefficients.

$$C_g \triangleq 10\log_{10}\frac{\sigma_x^2}{\left(\prod_{k=0}^{n-1}\sigma_{x_k}^2\|f_k\|^2\right)^{\frac{1}{n}}},$$

where

- n: the number of a transform basis function,
- σ_x^2: the input variance,
- $\sigma_(x_k)^2$: the variance of kth transform outputs,
- $\|f_k\|^2$: the norm of the k-th synthesis basis function.

Obtainable accuracy of the developed approximations of the DCT-16 (cont.)

Comparison to the original transform and to a state-of-art DCT approximation:

Transform	Coding gain
Floating-point $\mathrm{DCT}\text{-}16_2$	9.4555 [dB]
16-point binDCT[3]	9.4499 [dB]
Proposed algorithm (9 bit fractional part)	9.4546 [dB]
Proposed algorithm (12 bit fractional part)	9.4553 [dB]

Accuracy depends on the implementation bit width.

Close approximation even for moderate bit widths.

[3] J. Liang and T. D. Tran, "Fast multiplierless approximations of the DCT with the lifting scheme," *IEEE Trans. Signal Process.*, vol. 49, no. 12, pp. 3032–3044, Dec. 2001

Summary

A methodology for approximating and implementing the DCT-2 has been developed:

- obtained by combining the recent approaches:
  - the Algebraic Signal Processing (ASP)
  - the Algebraic Integer Encoding (AIE)
- handles any power-of-two size of the transform
- allows the original transform to be closely approximated
- results in regular signal flow graphs
- enables efficient multiplierless implementation (only bit shifts and additions)

The approach has been validated by developing an algorithm for the 16-point DCT.