Note that the proposed methodology for examining activity logs is more or less automatic - that is, it allows for little human interaction. However, experimental results have shown that such algorithms are not very successful. Therefore, it is worth exploring how human experts can participate in the different stages of pattern discovery. For example, when a new pattern is discovered, it can be examined by a domain expert before it is recorded in the pattern database.

To summarize, we propose a methodology for network activity log processing. We believe that the theoretical background behind the proposed methodology is sound and that when applied in practice the proposed network activity analyzer will produce results better than most existing network monitoring systems. However, the only way to find this for certain is to implement the proposed network activity analyzer and compare its performance and effectiveness to existing network monitoring systems.

Bibliography

[AgImSw93] Agrawal R., Imielinski T., Swami A., Mining association rules between sets of items in large databases. In *Proceeding of the ACM SIGMOD Conference on Management of Data*, pp. 207-216, 1993

[ChDa98] Chomicki, J., Toman D. Temporal Logic in Information Systems. In *Logics for Databases and Information Systems*, Kluwer Academic publishers, pp. 31-70, 1998

[Chom95] Chomicki, Jan. Efficient Checking of Temporal Integrity Constraints Using Bounded History Incoding. In *ACM Transactions on Database Systems*, Vol 20, No. 2, pp. 149-186, June 1995

[HaSuVi99] Hasan M., Sugla B., Viswanathan R. A Conceptual Framework for Network Management Event Correlation and Filtering Systems. In *Sixth IEEE/IFIP International Conference on Integrated Network Management*, May 1999

[LePaSt99] Lee W, Park C., Salvatore J., Automated intrusion detection using NFR: Methods and experiences. *Workshop on Intrusion Detection and Monitoring*, 1999

[LeFa93] Leinwand A, Fang K,. Network Management - A Practical Perspective. Addison Weley, 1993

[LiMoYa99] Liu G., Mok A.K., Yang E.J., Composite Events for Network Event Correlation, In *Sixth IFIP/IEEE International Symposium on Integrated Network Management*, May 1999

[MaToVe95] Mannila H., Toivonen H., Verkamo A., In *Proceedings of The First International Conference on Knowledge Discovery and Data Minning*, Montreal, Canada, pp. 20-21, 1995

Authors' Information

Lubomir Stanchev - Indiana University - Purdue University Fort Wayne, 2101 E. Coliseum Blvd., Fort Wayne, IN 46805 USA e-mail: stanchel@ipfw.edu

ON THE ERROR-FREE COMPUTATION OF FAST COSINE TRANSFORM

Vassil Dimitrov, Khan Wahid

Abstract: *We extend our previous work into error-free representations of transform basis functions by presenting a novel error-free encoding scheme for the fast implementation of a Linzer-Feig Fast Cosine Transform (FCT) and its inverse. We discuss an 8x8 L-F scaled Discrete Cosine Transform where the architecture uses a new algebraic integer quantization of the 1-D radix-8 DCT that allows the separable computation of a 2-D DCT without any intermediate number representation conversions. The resulting architecture is very regular and reduces latency by 50% compared to a previous error-free design, with virtually the same hardware cost.*

Keywords: *DCT, Image Compression, Algebraic Integers, Error-Free Computation.*

ACM Classification Keywords: *I.4.2 Compression (Coding), I.1.2 Algorithms, F.2.1 Numerical Algorithms.*

Introduction

The Discrete Cosine Transform (DCT) is the core transform of many image processing applications for reduced bandwidth image and video transmission, including the JPEG image processing standard and high performance video coding standards such as MPEG and H.263.

Because of the enormous popularity of the DCT, much research has been published on fast DCT algorithms [Arai, 1988][Duhamel, 1990][Feig, 1992][Linzer, 1991], where the effort is devoted to reducing the number of arithmetic operations used. Scaled DCT algorithms rely on a post-pointwise scaling operation which removes some of the arithmetic operations from the main transform computation. The post-scaling operations for a 2-D DCT can be delayed until the end of the transformation process. In this paper, we discuss a new algebraic-integer-mapping for the Linzer-Feig scaled-DCT [Linzer, 1991].

Several algorithms and architectures have previously been proposed to optimize both pure and scaled DCT implementations using 1-D and 2-D algebraic integer (AI) encoding of the DCT basis functions. Both single and multidimensional AI schemes have been used for low-complexity and parallel architectures [Dimitrov, 1998][Dimitrov, 2003][Wahid, 2004]. In most of these previous published encoding techniques, conversion from the AI output of each 1-D DCT computation has been required, even if the DCT is being used in a separable 2-D DCT computation. Here we introduce a new algebraic integer encoding technique which removes the need for conversion to binary at the end of the first 1-D DCT. We also extend this concept for 2-D error-free algebraic integer encoding and supply details on the computational complexity and mathematical precision required to implement the algorithm.

Background

For a real data sequence $x(n)$ of length *N*, the DCT is defined as follows:

$$F(k) = 2\sum_{n=0}^{N-1} x(n)\cos\left[\frac{(2n+1)k}{2N}\pi\right];\ 0 \le k \le N-1 \qquad (1)$$

The Inverse DCT (IDCT) is also defined as:

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} \overline{F}(k)\cos\left[\frac{(2n+1)k}{2N}\pi\right];\ 0 \le n \le N-1 \qquad (2)$$

Where, $\overline{F}(k) = \begin{cases} \dfrac{F(0)}{2} & k = 0 \\ F(k) & otherwise \end{cases}$

Both DCT and IDCT are separable transformations. So the 2-D transform can be computed by first performing 1-D transforms on each row and then performing 1-D transforms on each column.

Algebraic Integer Encoding

Algebraic integers are defined by real numbers that are roots of monic polynomials with integer coefficients [Dedekind, 1996]. As an example, let $\omega = e^{\frac{2\pi j}{16}}$ denote a primitive 16th root of unity over the ring of complex numbers. Then ω is a root of the equation $x^8 + 1 = 0$. If ω is adjoined to the rational numbers, then the associated ring of algebraic integers is denoted by $\boldsymbol{Z}[\omega]$. The ring $\boldsymbol{Z}[\omega]$ can be regarded as consisting of polynomials in ω of degree 7 with integer coefficients. The elements of $\boldsymbol{Z}[\omega]$ are added and multiplied as polynomials, except that the rule $\omega^8 = -1$ is used in the product to reduce the degree of powers of ω to below 8. For an integer, M, $\boldsymbol{Z}[\omega]_M$ is used to denote the elements of with coefficients between $-\frac{M}{2}$ and $\frac{M}{2}$.

Algebraic integer quantization has been used in DSP applications for about two decades and it has been demonstrated that it can be used for extremely efficient implementation of real-valued transforms such as the Discrete Hartley Transform [Baghaie, 2001], and the Discrete Wavelet Transform [Wahid, 2003].

Fast Cosine Transform Algorithm

The algorithm proposed by Linzer and Feig [Linzer, 1991] is presented as a signal flow graph in Figure 1, where $\{a_i\}$ are input elements, $\{S_i\}$ are scaled DCT coefficients, and fixed multipliers are given by (3):

$$c_4 = \cos\frac{4\pi}{16};\ t_1 = \tan\frac{\pi}{16};\ t_2 = \tan\frac{2\pi}{16};\ t_5 = \tan\frac{5\pi}{16} \tag{3}$$

The outlined area in Figure 1 (with a hardware cost of 10 multiplications and 10 additions) is where our new algebraic integer mapping will be used, and we will show that our mapping technique reduces the hardware complexity and also produces error-free transform outputs.

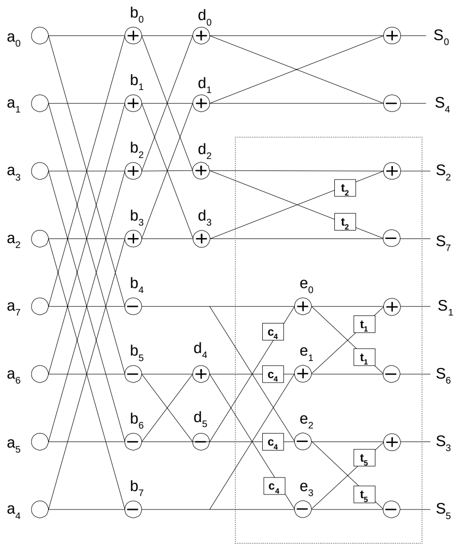

Figure 1: 1-D DCT (finite-precision binary)

1-D Algebraic Integer Encoding: Let $z = 2 + \sqrt{2+\sqrt{2}}$ and consider the polynomial expansion:

$$f(z) = a_0 + a_1 z + a_2 z^2 + a_3 z^3 \tag{4}$$

Now considering the expressions: $\cos\frac{4\pi}{16} = \frac{\sqrt{2}}{2}$, $\tan\frac{\pi}{16} = \frac{2}{\sqrt{2-\sqrt{2}}} - \sqrt{2} - 1$, $\tan\frac{2\pi}{16} = \sqrt{2} - 1$, and $\tan\frac{5\pi}{16} = \frac{2}{\sqrt{2+\sqrt{2}}} + \sqrt{2} - 1$, we can represent all the multipliers from eqn. (3) exactly with integer coefficients as shown in Table 1 (here, $\tilde{c}_4 = 2c_4$).

Table 1: 1-D error-free multiplier encoding

	a_0	a_1	a_2	a_3
$\tilde{c}_4$	2	-4	1	0
t_1	-7	14	-7	1
t_2	1	-4	1	0
t_5	1	-12	7	-1

Note that the multiplication between any real number and the coefficients in Table 1 can now be implemented with at most 4 shifts and 1 addition (for 14). This reduces the 10 multiplications to only 5 AI additions (actually, 23 shifts and 5 additions). So, the total number of additions required to perform the first 1-D DCT is 21. In the case of an 8x8 2-D DCT, we will need a total of 736 additions including the final substitution. We note that there is no longer any precision problem since the AI encoding provides an exact representation. The flow graph of Figure 1 can now be implemented as shown simply in Figure 2.

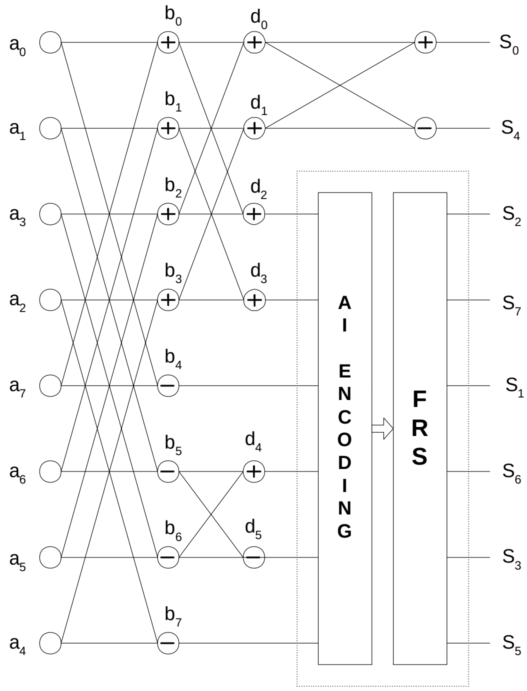

Figure 2: 1-D DCT (error-free AI encoding)

The real numbers of $f(z)$ form a ring which may be denoted as $Z[2+\sqrt{2+\sqrt{2}}]$. Addition in this ring is component-wise and multiplication is equivalent to a polynomial multiplication modulo $z^4 - 8z^3 + 20z^2 - 16z + 2 = 0$, which is demonstrated in Table 2.

Table 2: Multiplication in ring $Z[2+\sqrt{2+\sqrt{2}}]$

$\tilde{c}_4$	t_1	$\tilde{c}_4 . t_1$
(2,-4,1,0)	(-7,14,-7,1)	(-8,6,-1,0)

2-D Algebraic Integer Encoding: Applying a 2-D algebraic integer scheme to this algorithm results in a more sparse representation and more flexible encoding compared to previous techniques [Dimitrov, 2003]. For this encoding, the polynomial is expanded into 2 variables:

$$f(z_1, z_2) = \sum_{i=0}^{K} \sum_{j=0}^{L} a_{ij} z_1^i z_2^j \tag{5}$$

Here, we choose K=2 and L=2 to guarantee error-free encoding. For the most efficient encoding (i.e., to obtain the sparsest matrix), we have found the following: $z_1 = \sqrt{2+\sqrt{2}}$ and $z_2 = \sqrt{2-\sqrt{2}}$. The corresponding multiplier coefficients (including the cross-multipliers) are encoded in the form of $\begin{bmatrix} a_{00} & a_{10} & a_{20} \\ a_{01} & a_{11} & a_{21} \\ a_{02} & a_{12} & a_{22} \end{bmatrix}$ as shown in

Table 3. We have therefore, mapped the multiplier transcendental functions (cosine and tangent) without any error and with very low complexity. The implementation is same as shown in Figure 2.

Table 3: 2-D error-free multiplier encoding

$\tilde{c}_4$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	t_2	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$
t_1	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	t_5	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
$\tilde{c}_4 t_1$	$\begin{bmatrix} -2 & 0 & 0 \\ 2 & -1 & 0 \\ 0 & 2 & 0 \end{bmatrix}$	$\tilde{c}_4 t_5$	$\begin{bmatrix} 2 & -2 & 0 \\ 0 & -1 & 2 \\ 0 & 0 & 0 \end{bmatrix}$

From Table 3, we see that the coefficients are either 0's, 1's or 2's, so no additions and at best 1 shift is required to encode these numbers which reduces the total number of AI additions to perform the first 1-D transform to only 16 (496 in total for an 8x8 2-D DCT).

Final Reconstruction Step: In the final reconstruction stage (FRS), we map the AI numbers to a binary output. FRS is performed based on the precision in Table 4 and the flow graphs in Figure 3 and Figure 4. For the final reconstruction, we can use Horner's rule [Knuth, 1981]. In that case, eqn. (4) and eqn. (5) can be re-written as:

$$f(z) = ((a_3 z + a_2)z + a_1)z + a_0 \tag{6}$$

$$f(z_1, z_2) = a_{00} + z_1 z_2 (a_{11} + a_{21} z_1 + a_{12} z_2) \tag{7}$$

This final substitution stage generates some rounding errors but these errors are only introduced at the very end of the transformation process, not distributed throughout the calculation as is the case for a finite-precision binary implementation.

Table 4: FRS for different encoding scheme

Scheme	*Parameter*		*FRS*
1-D	$z = 2 + \sqrt{2 + \sqrt{2}}$	8 bits	$4 - 2^{-3} - 2^{-5}$
		12 bits	$4 - 2^{-3} - 2^{-5} + 2^{-7}$
2-D	$z_1 = \sqrt{2 + \sqrt{2}}$	8 bits	$2 - 2^{-3} - 2^{-5}$
		12 bits	$2 - 2^{-3} - 2^{-5} + 2^{-8}$
	$z_2 = \sqrt{2 - \sqrt{2}}$	8 bits	$1 - 2^{-2} + 2^{-6}$
		12 bits	$1 - 2^{-2} + 2^{-6} - 2^{-11}$

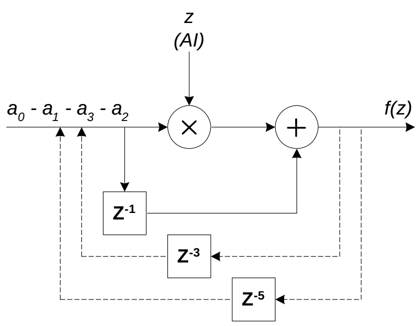

Figure 3: Final reconstruction step (1-D encoding)

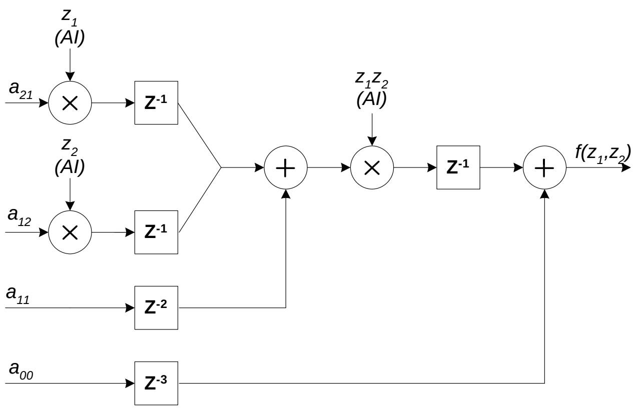

Figure 4: Final reconstruction step (2-D encoding)

For the scaled IDCT, the Linzer and Feig algorithm uses the same fixed multiplier coefficients given by eqn. (3). So, in this case, we can use the same encoding with the same precision described in the above section.

Comparisons

In Table 5, we compare the computational complexity of previously published AI-based DCT encoding with the proposed scheme. In all cases, the new 2-D AI encoding scheme has the least number of computations.

In Table 6, we present a comparison between some other published 2-D DCT architectures and the proposed algebraic integer approach. Taking the additions as the main computational block, the new multidimensional algebraic-integer-quantization based architecture clearly has the lowest hardware count, particularly considering that all the AI computations are performed without any error.

Table 5: Hardware complexity for different AIQ schemes

Algorithm	Degree of Polynomial	Additions	Shifts	Total Additions
1-D AI-based Chen DCT [Dimitrov, 1998]	7	6	9	156
2-D AI-based Chen DCT [Dimitrov, 2003]	3	3	4	132
AI-based Arai [Wahid, 2004]	3	1	5	30
Proposed 1-D AIQ	3	5	23	32
Proposed 2-D AIQ	2	0	6	29

Table 6: Comparison between different 8-point 2-D DCT

Algorithm	Multiplications	Additions
Linzer-Feig sDCT [Linzer, 1991]	160	416
DCT-SQ [Arai, 1988]	80	464
Distributed DCT [Shams, 2002]	0	672
Proposed 1-D AIQ	0	736
Proposed 2-D AIQ	0	496

Conclusions

In this paper, we have introduced a new encoding scheme to compute both 1-D and 2-D Fast Cosine Transform IFCT by error-free mapping of transcendental functions. This new quantization technique effectively reduces the overall number of arithmetic operations, and allows a multiplication-free, parallel, and very fast hardware implementation. Except for the final reconstruction stage, the complete 2-D DCT and IDCT can be implemented without error (infinite precision). This idea of using an algebraic integer quantization scheme can be easily generalized to other algorithms when it is necessary to use real algebraic numbers of special form. Our future work is directed towards the VLSI implementation of this approach for 2-D DCT and IDCT.

Acknowledgements

The authors also wish to thank the Alberta Informatics Circle of Research Excellence (iCORE), and the Natural Sciences and Engineering Council (NSERC) of Canada for their financial support for this research. The authors are also indebted to the Canadian Microelectronics Corporation (CMC) for providing the hardware and software infrastructure used in the development of this design.

Bibliography

[Arai, 1988] Y. Arai, T. Agui and M. Nakajima, “A Fast DCT-SQ Scheme for Images”, Transactions of Institute of Electronics, Information and Communication Engineers, vol. E71, no. 11, pp.1095-1097, 1988.

[Baghaie, 2001] R. Baghaie and V. Dimitrov, “Systolic Implementation of Real-valued Discrete transforms via Algebraic Integer Quantization”, An International Journal on Computers and Mathematics with Applications, vol. 41, pp.1403-1416, 2001.

[Dimitrov, 1998] V. S. Dimitrov, G. A. Jullien and W. C. Miller, “A New DCT Algorithm Based on Encoding Algebraic Integers”, IEEE International Conference on Acoustics, Speech and Signal Processing, pp.1377-1380, 1998.

[Dimitrov, 2003] V. Dimitrov and G. A. Jullien, “Multidimensional Algebraic Integer Encoding for High Performance Implementation of the DCT and IDCT”, IEE Electronics Letters, vol. 29, no. 7, pp.602-603, 2003.

[Dedekind, 1996] Richard Dedekind, “Theory of Algebraic Integers”, Translated and introduced by John Stillwell, 1996.

[Duhamel, 1990] P. Duhamel and C. Guillemot, “Polynomial Transform Computation of the 2-D DCT”, IEEE International Conference on Acoustics, Speech and Signal Processing, pp.1515-1518, 1990.

[Feig, 1992] E. Feig and S. Winograd, “Fast Algorithms for the Discrete Cosine Transform”, IEEE Transactions on Signal Processing, vol. 40, no. 9, pp.2174-2193, 1992.

[Knuth, 1981] D. Knuth, “The Art of Computer Programming”, vol. 2 - Seminumerical Algorithms, 3rd edition, Addison Wesley, 1981.

[Linzer, 1991] E. Linzer and E. Feig, “New Scaled DCT Algorithms for Fused Multiply/Add Architectures”, IEEE International Conference on Acoustics, Speech and Signal Processing, pp.2201-2204, 1991.

[Shams, 2002] A. Shams, W. Pan, A. Chidanandan and M. Bayoumi, “A Low Power High Performance Distributed DCT Architecture”, Proceedings of IEEE Annual Symposium on VLSI, pp.21-27, 2002.

[Wahid, 2003] K. Wahid, V. Dimitrov, G. Jullien and W. Badawy, “Error-Free Computation of Daubechies Wavelets for Image Compression Applications”, IEE Electronics Letters, vol. 39, no. 5, pp.428-429, March 2003.

[Wahid, 2004] Vassil Dimitrov, Khan Wahid and Graham Jullien, “Multiplication-Free 8x8 2D DCT Architecture using Algebraic Integer Encoding”, IEE Electronics Letters, vol. 40, no. 20, pp.1310-1311, 2004.

Authors’ Information

Vassil Dimitrov – Dept. of Electrical and Computer Engineering, University of Calgary, Calgary, Alberta, Canada; e-mail: dimitrov@atips.ca

Khan Wahid – Dept. of Electrical and Computer Engineering, University of Calgary, Calgary, Alberta, Canada; e-mail: wahid@atips.ca