Octonion CORDIC Algorithms for DSP

Evgueni I. Doukhnitch
Eastern Mediterranean University
Evgueni.doukhnitch@emu.edu.tr

Abstract. New, efficient, 8-D rotation CORDIC-like algorithms for matrix computations are presented. These algorithms are a natural extension to 8 dimensions of Quaternion CORDIC algorithms, offered by J.M. Delosme and S.F. Hsiao. This leads to significantly reduced computations in contrast to 8-D Householder CORDIC algorithms given by the same authors. Such algorithms may be utilized for designing VLSI array processors or configurable FPGA-s for linear systems, the eigenvalues, singular values, least squares and other linear algebra problems in DSP.

Index Terms- CORDIC, computer arithmetic, multi-dimensional rotations, matrix computations, VLSI.

1. Introduction

The very fast growth of modern VLSI complexity offers a hardware realization of an ever-growing share of mathematical means. It essentially raises the computer performance. However, known multidimensional algorithms for signal processing are not hardware-oriented. The exceptions are the famous CORDIC-algorithms [1] and some FFT-algorithms [2]. New algorithms are needed to satisfy VLSI-technology requirements. Main requirements are the following:

- algorithms must have a guaranteed accuracy and convergence after a fixed number of steps;
- every step of the algorithm must have a limited set of simple operations with the same realization time;
- algorithms must have the possibility of decomposition on equal parts with a limited set of types;
- algorithms must realize the highest possible typical computing procedures which are frequently found in signal processing methods.

For ensuring the previous requirements it is necessary to pick out a set of large operations (macro-operations) for programming of linear algebra problems and to design special algorithms for fast implementation. It offers the realization of a computation process by a configurable FPGA, for example, using one or several VLSI-chips.

2. Discrete Linear Transforms

The analysis of mathematical content of signal processing shows the main problems are linear systems. They are system of equations, eigenvalues, singular values, least squares, and other problems. The typical matrix operation for their solutions is an one-sided linear transformation:

$$\mathbf{Y}=\mathbf{PX}. \qquad (1)$$

Sometimes it's possible to perform the factorization of matrix **P** by simple matrices:

$$\mathbf{P}=\prod_{\mathbf{i}=0}^{\infty}\mathbf{P}_{\mathbf{i}}\,. \qquad (2)$$

Hence, the one-side transformation (1) may be realized by an iteration process:

$$\mathbf{Y}_{\mathbf{i+1}} = \mathbf{P}_{\mathbf{i}}\mathbf{Y}_{\mathbf{i}}, \qquad (3)$$

where i = 0, 1, 2, …, n; $\mathbf{Y}_0 = \mathbf{X}$. The result is $\mathbf{Y}_n \to \mathbf{PX}$, if $n \to \infty$.

This process is a ***discrete linear transform*** (DLT). When elements of matrices $\mathbf{P}_i$ are ones, zeros or twos with integer power this process can be very fast implemented by hardware. For example, Givens rotation is realized by the well-known hardware-oriented CORDIC-algorithms for two-dimensional plane rotation.

The transformation matrix is $\mathbf{P}=\begin{pmatrix}\cos\alpha & \sin\alpha\\ -\sin\alpha & \cos\alpha\end{pmatrix}$. Hardware - oriented algorithm of DLT uses the matrix **P** as a matrix product of elementary rotations $k\mathbf{P}=\prod_{\mathbf{i}=0}^{\mathbf{n}}\mathbf{R}_{\mathbf{i}}$, where $\mathbf{R}_{\mathbf{i}}=\begin{pmatrix}1 & \xi_i 2^{-i}\\ -\xi_i 2^{-i} & 1\end{pmatrix}$ – rotation matrix of the i-th iteration; $\xi_i=\pm 1$ – operator of a rotation direction; $k=\prod_{i=0}^{n}1/\cos\Delta\varphi_i=\prod_{i=0}^{n}(1+2^{-2i})^{1/2}$ – coordinates lengthening (scaling) factor; n-approximate number of bits for result representation. The transformation (1) converts a vector $\mathbf{X}$ to $k\mathbf{Y} = \left(\prod_{i=0}^{n} \mathbf{R_i}\right)\mathbf{X}$, and the algorithm for its implementation is:

$$\left.\begin{array}{l} \mathbf{Y_{i+1}} = \mathbf{R_i Y_i};\quad \varphi_{i+1} = \varphi_i - \xi_i \Delta\varphi_i;\quad \Delta\varphi_i = \mathrm{arctg}2^{-i}; \\ \xi_i = sign\varphi_i;\quad i = \overline{0, \mathrm{n}};\quad \varphi_0 = \alpha;\quad \mathbf{Y_0} = \mathbf{X} \end{array}\right\}$$

(4)

It involves only simple operations of addition and shift thus differing by a simplicity and high speed in hardware implementation. They are suitable for fast software realization by RISC-processors also. The VLSI-realization of the CORDIC-processor is often utilized for DSP [2]. The dimensionality of CORDIC-algorithms is equal to two. This is a main disadvantage. The long sequence of these macro-operations is required for problem solving.

3. Octonion DLT-s

The widespread way for solving linear algebra problems is zeroing entries in a vector or a matrix by sequence of rotations or reflections. For CORDIC algorithm (4) ξ_i=-sign(y_i) is taken to zero element x_2 of 2-D vector $\mathbf{X}$. It seems reasonable to speed up matrix computations by expressing them in terms of higher dimensional rotations. Householder reflections are exceedingly useful for introducing zeros on a grand scale, e.g. the annihilation of all but the first component of a vector. There are two outstanding multidimensional CORDIC-like algorithms suggested by J.M. Delosme and S.F. Hsiao for VLSI-implementation of these transformations. To generalize the original CORDIC algorithms they offered the Quaternion or Pseudo-quaternion CORDIC algorithms (QCA) [3], [8] for 4-D rotations:

$$\mathbf{Y_{i+1}} = \mathbf{R_{4,i}}\, \mathbf{Y_i},\ (0 \le i \le n;\ \mathbf{Y_0}=\mathbf{X}),$$

where the elementary rotation matrix is

$$\mathbf{R_{4,i}} = \begin{pmatrix} 1 & \alpha_i t_i & \beta_i t_i & \gamma_i t_i \\ -\alpha_i t_i & 1 & -\gamma_i t_i & \beta_i t_i \\ -\beta_i t_i & \gamma_i t_i & 1 & -\alpha_i t_i \\ -\gamma_i t_i & -\beta_i t_i & \alpha_i t_i & 1 \end{pmatrix}$$

The rotation parameters t_i are equal to $2^{-f(i)}$ with {f(i)} a non-decreasing positive integer sequence and the control signs $\alpha_i \div \rho_i$ are either 1 or –1. To bring real 4-D vector $\mathbf{X}$ along the first canonical axis control signs are selected according to

$$\alpha_i = f_i \cdot \mathrm{sign}(y_{2,i});\qquad \beta_i = f_i \cdot \mathrm{sign}(y_{3,i});$$

$$\gamma_i = f_k \cdot \mathrm{sign}(y_{4,i});\qquad f_i = \mathrm{sign}(y_{1,i}).$$

For m dimensions they presented the Householder CORDIC algorithms (HCA), which have the limits for practical values of m "to less than 10 or so" [4] because the complexity in terms of area and computation time grows slightly faster than linearly. From the practical point of view m=8 may be taken. In this case it's possible to design 8-D CORDIC-like algorithms those are simpler than HCA and are suitable for solving the same linear algebra problems.
Just as a complex number can be written as an ordered pair of real numbers, a quaternion can also be written as an ordered pair of complex numbers. Following this approach one is then naturally led to consider ordered pairs of quaternions – these new objects being called Cayley numbers or octaves or ***octonions*** (8-dimensional objects) [5]. In a way similar to QCA in Euclidean case new algorithm is called 8-D Octonion CORDIC algorithm since its elementary rotation matrices

$$\mathrm{R}_{8,i} = \begin{pmatrix} 1 & \alpha_i t_i & \beta_i t_i & \gamma_i t_i & \delta_i t_i & \lambda_i t_i & \mu_i t_i & \rho_i t_i \\ -\alpha_i t_i & 1 & -\delta_i t_i & -\rho_i t_i & \beta_i t_i & -\mu_i t_i & \lambda_i t_i & \gamma_i t_i \\ -\beta_i t_i & \delta_i t_i & 1 & -\lambda_i t_i & -\alpha_i t_i & \gamma_i t_i & -\rho_i t_i & \mu_i t_i \\ -\gamma_i t_i & \rho_i t_i & \lambda_i t_i & 1 & -\mu_i t_i & -\beta_i t_i & \delta_i t_i & -\alpha_i t_i \\ -\delta_i t_i & -\beta_i t_i & \alpha_i t_i & \mu_i t_i & 1 & -\rho_i t_i & -\gamma_i t_i & \lambda_i t_i \\ -\lambda_i t_i & \mu_i t_i & -\gamma_i t_i & \beta_i t_i & \rho_i t_i & 1 & -\alpha_i t_i & -\delta_i t_i \\ -\mu_i t_i & -\lambda_i t_i & \rho_i t_i & -\delta_i t_i & \gamma_i t_i & \alpha_i t_i & 1 & -\beta_i t_i \\ -\rho_i t_i & -\gamma_i t_i & -\mu_i t_i & \alpha_i t_i & -\lambda_i t_i & \delta_i t_i & \beta_i t_i & 1 \end{pmatrix}$$

(5)

are real matrix representations of particular *k_i-length octonions* ($k_i = \sqrt{1+7t_i^2}$). The rotation parameters t_i are equal to $2^{-f(i)}$ with {f(i)} a non-decreasing positive integer sequence and the control signs $\alpha_i \div \rho_i$ are either 1 or –1. Without performing the scaling by k_i^{-1} the implementation of one elementary rotation consist of eight concurrent shift-and-add operations. The known means may be used for decomposition of overall scaling factor $k = \prod_{i=1}^{n} (1+7t_i^2)^{-1/2}$ into several simple factors such that the multiplication of 8-D real vector by each of those factors is also implemented by eight concurrent shift-and-add operations, e.g. [4]. For fixed bit accuracy, the total number of iterations, including unscaled and scaled iterations, in one 8-D octonion CORDIC operation is about the same as that in 2-D CORDIC operation. For example, for 32-bits accuracy only three additional iterations are needed for algorithm convergence and five – for scaling multiplication.
The Octonion CORDIC algorithm

$$\mathbf{Y_{i+1}} = \mathbf{R_{8,i}}\, \mathbf{Y_i},\ (0 \le i \le n;\ \mathbf{Y_0}=\mathbf{X}), \qquad (6)$$

as well as original CORDIC algorithm, can be used in two modes - *rotation* (application of transformation) and *vectoring* (evaluation of a parameter of transformation). Usual task of vectoring in such algorithms is annihilation of required components in a vector given. As after plane Volder's rotation one of two components is zero, in 8-D space seven components may be zeros. A sequence of elementary rotations with matrices (5) is applied to a 8-D vector $X = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]^T$ to bring it along the first canonical axis. To achieve this, the control signs are selected for (6) according to

$$\left.\begin{array}{ll} \alpha_i = f_i \cdot \mathrm{sign}(y_{2,i}); & \beta_i = f_i \cdot \mathrm{sign}(y_{3,i}); \\ \gamma_i = f_k \cdot \mathrm{sign}(y_{4,i}); & \delta_i = f_i \cdot \mathrm{sign}(y_{5,i}); \\ \lambda_i = f_i \cdot \mathrm{sign}(y_{6,i}); & \mu_i = f_k \cdot \mathrm{sign}(y_{7,i}); \\ \rho_i = f_i \cdot \mathrm{sign}(y_{8,i}); & (0 \le i \le n;\ \mathbf{Y_0 = X}), \end{array}\right\} \quad (7)$$

where $f_i = \mathrm{sign}(y_{1,i})$; $y_{j,i}$ denotes the j-th component of vector $\mathbf{Y_i}$ at the beginning of the (i+1)-th iteration. In rotation mode, the predetermined control signs are used to rotate a 8-D vector. They can be obtained from a preliminary vectoring operation in-fly.

The hardware implementation of Octonion CORDIC algorithm is similarly to quaternion CORDIC-processor [3]. The major components are 8 shifters that perform right-shift by i bit positions, 8 carry-save-adders that transform 8 operands into 2, 8 adders and 8 storage registers. The proposed processor can calculate an 8-D rotation in roughly the same time as standard 2-D CORDIC-processor calculates plane rotation because the latency distinction is one carry-save-addition only. In contrast to 8-D Householder CORDIC processor the shifters for multiplications by t_i^2 are not necessary thus significantly reducing the hardware cost.

4. Computational Structures for Matrix Triangularization

A DLT with matrix $\mathbf{P} = k^{-1} \prod_{\mathbf{i=0}}^{\mathbf{n}} \mathbf{R_{8,i}}$ is orthogonal transformation. Therefore it can be used to implement a lot of important matrix methods. For example, to triangularize a given 8x8 matrix **A** we can use the Octonion CORDIC algorithm:

$$\mathbf{A^{(i+1)} = R_{8,i}\, A^{(i)}}, \quad (0 \le i \le n;\ \mathbf{A^{(0)} = A}) \quad (8)$$

If control signs are determined by (7) for vector $\mathbf{Y_i} = \mathbf{A_1^{(i)}} = [a_{11}^{(i)}, a_{21}^{(i)}, a_{31}^{(i)}, a_{41}^{(i)}, a_{51}^{(i)}, a_{61}^{(i)}, a_{71}^{(i)}, a_{81}^{(i)}]^T$, the result of this transformation is:

$$\mathbf{A^{(n)}} = \begin{pmatrix} a'_{11} & a'_{12} \dots a'_{18} \\ 0 & a'_{22} \dots a'_{28} \\ & \dots \\ 0 & a'_{82} \dots a'_{88} \end{pmatrix} \quad (9)$$

The computational process can be organized like Householder method. After 7 steps of similar transformations for vectors $\mathbf{X} = \mathbf{A}_j = [a_{jj}, a_{j+1,j}, \dots, a_{8j}]$ (j=1,…,7) the overtriangular matrix will be received. The missed (j-1) components for vectors **X** are equal to zero.

For pipeline processing n sets of 8 units are required to compute matrix (9) at all steps (i=0,1,2,…,n) .Units connection is shown in Figure. Every unit $U_r^{(i)}$ (r=1,2,…,8) performs one iteration of algorithms (6) and (7). It has inputs c_r for operands of operations (7) and inputs a_r for operations (6). A hardware for scaled iterations is not shown. The macropipeline with 7 such sets may be utilized for the computation of an overtriangular matrix. Units number for j-th set (stage of macropipeline) is less by (j-1) units.

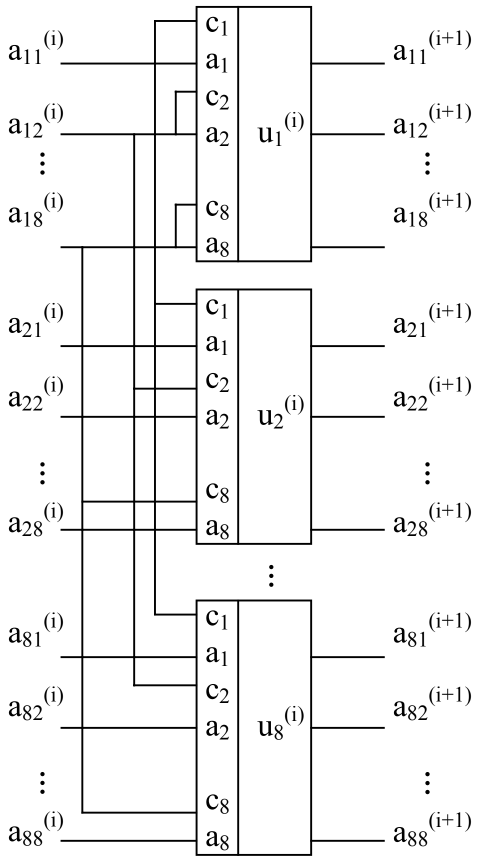

Fig. Connection of units for i-th iteration of (8).

So, a throughput of one matrix triangularization per clock period may be achieved using described "doubly piped"[2] Octonion CORDIC modules. For arbitrary size matrix N×N (N>8) we can use partitioned matrix algorithms [2].

5. Conclusions

New CORDIC-like algorithms were offered for multi-dimensional rotations. These results significantly reduce the computation time of important matrix algorithms and significantly reduce the hardware cost. Moreover, suggested technology may be used for designing new CORDIC-like algorithms [7].

It's necessary to mention that all improvements over the basic CORDIC algorithm (scaling iterations, redundant arithmetic, high-radix arithmetic, etc.) [6] may be incorporated in the offered algorithms also.

References

[1] J.E Volder, "The CORDIC Trigonometric Computing Technique." *IEEE Trans. On Electronic Computers,* vol. EC-9, pp. 227-231, 1960.

[2] S. Y. Kung, *VLSI Array Processors*, Prentice Hall, 1991.

[3] S.-F.Hsiao, J.-M. Delosme, "Parallel Processing of Complex Data Using Quaternion and Pseudo-Quaternion CORDIC Algorithms." *In Proceedings of the ASAP'94 Conf,* University of California, pp.125-130, 1994.

[4] S.-F.Hsiao, J.-M. Delosme, "Householder CORDIC Algorithms." *IEEE Trans. on Computers,* Vol. 44, No.8, pp. 900-1002, 1995.

[5] J.P. Ward, *Quaternions and Cayley Numbers*, Kluwer Acadmic Publishers, 1997.

[6] S.-F.Hsiao, C.-Lau and J.-M. Delosme, "Redundant Constant-Factor Implementation of Multi-dimensional CORDIC and Its Application to Complex SVD", *Journal of VLSI Signal Processing,* Vol.25, No.2, pp.155-166, June 2000.

[7]E.I.Doukhnitch, "Highly Parallel Multi-dimensional CORDIC-like Algorithms", *Proc.Int.Conf. Intelligent Multiprocessor Systems-2001*, pp. 44-48, Sept. 1999.

[8] S.-F.Hsiao, J.-M. Delosme, "Parallel Singular Value Decomposition of Complex Matrices Using Multi-dimensional CORDIC Algorithms", *IEEE Trans. on Signal Processing,* Vol. 44, No.3, pp. 685-697, 1996.