ANGLE RECODING METHOD FOR EFFICIENT IMPLEMENTATION OF THE CORDIC ALGORITHM

Yu Hen Hu

Dept. of Electrical & Computer Engineering
University of Wisconsin - Madison
Madison, WI- 53706

S. Naganathan

Electrical Engineering Department
Southern Methodist University
Dallas, TX- 75275

ABSTRACT

CORDIC (COordinate Rotation DIgital Computer) is an iterative arithmetic algorithm for computing generalized vector rotations without performing multiplications. For applications where the angle of rotation is known in advance, we shall present in this paper a method to speedup the execution of the CORDIC algorithm by reducing the total number of iterations. This is accomplished by using a technique called *Angle Recoding*, which encodes the desired rotation angle as a linear combination of very few elementary rotation angles. Each of these elementary rotation angle takes one CORDIC iteration to compute. The fewer the number of elementary rotation angles, the fewer the number of iterations are required. A Greedy algorithm which takes only $O(n^2)$ operations is developed in this paper to perform CORDIC angle recoding. It is proved that this algorithm is able to reduce the total number of required elementary rotation angles by at least 50 percent without affecting the computational accuracy. Simulation results will also be presented.

I. INTRODUCTION

A majority of fast signal processing algorithms are computationally intensive involving a large number of multiplications and computation of elementary functions. Many digital signal processing algorithms such as fast Fourier transform (FFT), Chirp Z transform (CZT), etc., involve a (circular) rotation operation [1-5] of the following form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{1}$$

where the vector[1] $[x\ y]^t$ is rotated through an angle θ to a new position $[x'\ y']^t$. In applications such as FFT and CZT, the angle θ is known prior to computation. Hence, the trigonometric functions $\cos\theta$ and $\sin\theta$ can be evaluated and stored in advance. During computation, these pre-stored values will be retrieved and multiplied to x and y. This requires four (real) multiplications and two additions. When millions of rotation operations are to be executed in real time for digital signal processing, and matrix related computations, special purpose arithmetic algorithms and hardwares are needed to speedup the computation.

The CORDIC [6-7] algorithm is a rotation based computing kernel suitable for performing plane rotation operations. In CORDIC, the rotation through the angle θ is accomplished by a sequence of successive rotations through n elementary rotation angles $\{a(i),\ i = 0, n-1\}$. Each a(i) is selected such that $\tan[a(i)] = 2^{-i}$ for some integer i. As a result, rotation through an elementary angle a(i) requires only shift and add operations. To be more specific in using the CORDIC algorithm, the given angle θ is decomposed as

$$\theta = \sum_{i=0}^{n-1} u(i)\, a(i) + \varepsilon \tag{2}$$

where ε is an angle approximation error such that $|\varepsilon| < a(n-1)$ and is negligible in practical computation [7]. The direction of rotation through a(i) is dictated by u(i) which takes a value of either + 1 or -1. In the CORDIC algorithm, these u(i)'s are computed from the angle iteration ($z(0) = \theta$):

$$z(i+1) = z(i) - u(i)a(i), \qquad i = 0, 1, \ldots, n-1.$$
$$u(i) = \textit{sign of } z(i)$$

Once the u(i)'s are determined, the remaining rotation operations are performed with the following iterations :

Initiation: $x(0) \equiv x,\ y(0) \equiv y$
For i = 0 to n-1 Do
CORDIC Rotation :

$$\begin{bmatrix} x(i+1) \\ y(i+1) \end{bmatrix} = \begin{bmatrix} 1 & \tan u(i)a(i) \\ -\tan u(i)a(i) & 1 \end{bmatrix} \begin{bmatrix} x(i) \\ y(i) \end{bmatrix}$$

Scaling (norm correction) operation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \prod_{i=0}^{n-1} \cos u(i)a(i) \cdot \begin{bmatrix} x(n) \\ y(n) \end{bmatrix}$$

*This research work is supported by National Science Foundation under contract DCI-8609283, MIP-8896111.

[1] In this paper $[\,]^t$ denotes vector transpose.

Note that $[x(n)y(n)]^t$ can be computed via n shift-and-add operations. The multiplication by $\prod_{i=1}^{n-1} \cos u(i)a(i) \equiv \frac{1}{K(n)}$ is an intrinsic *norm correction* operation[2] which is an overhead in computation. Fortunately, this norm correction constant is a known constant once the set of $\{u(i)a(i);\ i = 0, n-1\}$ are determined. As a result, multiplier recoding method, such as the modified Booth's algorithm [8] can be applied to speedup computation.

II. CORDIC ANGLE RECODING PROBLEM

As mentioned earlier, in conventional CORDIC algorithms, $u(i) = \pm 1$. Hence n CORDIC iterations will always be required even if $\theta = 0$, because each time $u(i) = 1$, or -1 a CORDIC iteration is to be computed (using one shift-and-add operation). In this paper, we propose to relax this constraint by allowing $u(i) = 0$. This would be advantageous in applications where θ is known in advance [5]. If u(i) can take ± 1 or 0, it would be desirable to minimize $\sum_{i=0}^{n} |u(i)|$ so that the total number of CORDIC iterations can be reduced. We shall call this technique **Angle Recoding** since it is similar to the multiplier recoding method employed in modern multiplier design. Now, the angle recoding problem can be formally stated:

The CORDIC Angle Recoding Problem : Given $\{a(i)$, for $i= 0, n-1\}$, and an angle θ, find $\{u(i); i= 0$ to $n-1$, $u(i) = 0, \pm 1\}$ such that

(i) $\theta = \sum_{i=0}^{n-1} u(i)\, a(i) + \varepsilon$ for $\varepsilon < a(n-1)$, and

(ii) $\sum_{i=0}^{n-1} |u(i)|$ is minimized.

For convenience, it will further be assumed that $|\theta| < 2a(0) = \pi/2$. This assumption may be ensured by applying the following procedures:

a. If $\theta > 2\pi$, replace θ by $\theta \bmod 2\pi$.

b. If $2\pi \geq \theta > \pi$, replace $[x\ y]^t$ by $[-x\ -y]^t$, and θ by $\theta - \pi$.

c. If $\pi \geq \theta > \pi/2$, replace $[x\ y]^t$ by $[y\ -x]^t$, and θ by $\theta - \pi/2$.

III. A GREEDY ALGORITHM

An optimum solution to the CORDIC angle recoding problem will have to test up to $O(3^n)$ combinations of $\{a(i)\}$ which would be impractical to evaluate for large value of n. In this paper, a Greedy Algorithm which takes only $O(n^2)$ operations is proposed. Later, it will be shown that the total number of elementary angles needed to represent θ, when this algorithm is used, will be less than n/2. Now, the algorithm is given below:

CORDIC Angle Recoding Algorithm:

Initialization: $\theta(0) = \theta$, $\{u(i) = 0;\ 0 \leq i \leq n-1\}$, $k = 0$.

Repeat until $|\theta(k)| < a(n-1)$ Do

1. Choose i_k, $0 \leq i_k \leq n-1$ such that
$$||\theta(k)| - a(i_k)| = \min_{0 \leq i \leq n-1} ||\theta(k)| - a(i)| \qquad (3)$$

2. $\theta(k+1) = \theta(k) - u(i_k)\, a(i_k)$ where, $u(i_k) = \mathrm{sign}(\theta(k))$

This is a *greedy* algorithm because at every step, it tries to represent the remaining angle (to be rotated) using a closest elementary CORDIC angle. Without looking ahead of future steps, this choice is the most reasonable one at the current iteration. To see that this algorithm actually converges, one would establish that $\theta(k)$ is a monotonically decreasing sequence. Since

$$|\theta(k+1)| = |\theta(k) - u(i_k)a(i_k)| = ||\theta(k)| - a(i_k)|$$

Thus, from (3), and the fact that[3] $|\theta(k)| > a(n-1)$

$$|\theta(k+1)| \leq ||\theta(k)| - a(n-1)| < |\theta(k)|$$

In what follows, it will be shown that if the algorithm terminates at $k = k^*$, $k^* < n/2$. For convenience, define $g(i) = a(i) - a(i+1)$ for $i= 0,1, \ldots, n-2$, where $a(i) = \tan^{-1}2^{-i}$, then

Lemma 1.

(a) $g(i) > 0$
(b) $a(i+2) < g(i) < a(i+1)$
(c) $g(i) > g(i+1)$

Proof: First, note that $a(i+1) < a(i)$, hence (a) is proved. Next, note that $a(i) < 2a(i+1)$ since $\tan(a(i)) = 2\tan(a(i+1))$, and that $\tan\theta$ is a concave function for $\theta > 0$. Thus, $g(i) - a(i+1) = a(i) - 2a(i+1) < 0$. To see that $g(i) > a(i+2)$, one has,

$$\tan g(i) - \tan a(i+2) = \tan(a(i) - a(i+1)) - \tan a(i+2)$$

$$= \frac{2^{-i} - 2^{-(i+1)}}{1 + 2^{-(2i+1)}} - 2^{-(i+2)} > 0$$

[2] The purpose of this norm correction operation is to preserve the norm of the vector $[x(i)\ y(i)]^t$ after rotation.

[3] Otherwise, the angle recoding procedure can be terminated.

Department of Electrical and Computer Engineering and
The Coordinated Science Laboratory

Since $\tan\alpha$ is a monotonic increasing function for $-\frac{\pi}{2} < \alpha < \frac{\pi}{2}$, (b) is proved. Now that

$$g(i)-g(i+1)=a(i)+a(i+2)-2a(i+1)$$

But it can easily be verified that

$$\tan(a(i)+a(i+2))-\tan(2a(i+1)) = \frac{2^{-(2i+2)}}{1-2^{-(2i+2)}} > 0$$

Again, the monotonic increasing property of the tangent function ensures that $g(i) > g(i+1)$.

Q.E.D.

Based on lemma 1, the following theorem is proved:

Theorem 1. Using the proposed CORDIC angle recoding algorithm, if $|\theta| \le a(0) = \pi/4$, then[4]

$$\sum_{i=0}^{n-1} |u(i)| \le n/2 \qquad (5)$$

Proof: Our strategy is to show that $i_k \ge 2k$. Since $a(n-1) < |\theta| < a(0)$, $|\theta|$ must fall between the intervals $[a(i_0+1), a(i_0)]$ or $[a(i_0), a(i_0-1)]$. Hence,

$$\begin{aligned}|\theta(1)| &= ||\theta(0)|-a(i_0)| \\ &< \text{Max.}\{g(i_0)/2,\ g(i_0-1)/2\} \\ &< g(0)/2 < a(1)/2 < a(2)\end{aligned}$$

Thus, the case of k= 1 is proved. By induction, assuming that

$$|\theta(k)| < a(2k) \qquad (6)$$

then

$$\begin{aligned}|\theta(k+1)| &< g(2k)/2 < a(2k+1)/2 < a(2k+2) \\ &= a(2(k+1))\end{aligned}$$

Hence, (6) is proved. When $2k \ge n-1$, the angle recoding procedure will be terminated. Thus, the total number of iterations is

$$k^* = \begin{cases} (n-1)/2 & n \text{ odd;} \\ n/2 & n \text{ even.} \end{cases}$$

Q.E.D.

IV. SIMULATION RESULTS AND DISCUSSIONS

To test the effectiveness of the proposed CORDIC angle recoding scheme, two simulations for both n = 16 and n = 32 are conducted. 4000 rotation angles (θ) uniformly distributed between 0 and a(0) are generated and encoded with the proposed scheme. For each of these test angles, after $\{u(i);\ i = 0 \text{ to } n-1\}$ are determined, $\sum |u(i)|$ is calculated. Also, the corresponding scaling factor

$$\prod_{i=0}^{n-1} \cos u(i)a(i) = 1/K(n)$$

for each angle is calculated. Using a modified Booth's (multiplier) recoding representation for 1/K(n), the total number of 1's and −1's in this representation is taken to be the additional iterations needed for CORDIC scaling (norm correction) operation. The total number of iterations for both CORDIC rotation operations and the norm correction operation is also calculated.

The distribution of k* and the total number of iterations (including the norm correction operations) is depicted in figure 1 for the case n= 16 and in figure 2, the case of n= 32 is depicted. The maximum, mean and the standard deviation of the distribution of k* for performing only the CORDIC rotation operations are given in Table I. The maximum number of k* for n= 16 is 7, and for n= 32 it is 14. The corresponding values of the distribution of k* for the total computation which includes both the rotation and scaling operations are given in Table II. The maximum value k* = 14 for n= 16 and k* = 28 for n= 32.

TABLE I (CORDIC ROTATION OPERATIONS)

	n= 16	n= 32
Max. number of iterations	7	14
Mean	4.959	10.28
Standard deviation	1.052	1.505

TABLE II (ROTATION & SCALING OPERATIONS)

	n= 16	n= 32
Max. number of iterations	14	28
Mean	9.626	20.04
Standard deviation	2.122	3.004

The proposed algorithm for *Angle Recoding* optimizes the number of CORDIC iterations and improves the speed of computation. In the VLSI implementation of the CORDIC algorithm, the area of the shifter can be reduced, with the optimized number of shift-and-add operations achieved through the proposed algorithm.

[4] For $\pi/4 < |\theta| < \pi/2 = 2a(0)$, one may rotate through a(0) first. The remaining angle then satisfies the assumption in this theorem. If this is the case, the total number of iterations will be increased by 1.

FIGURE 1. ANGLE RECODING (N = 16)

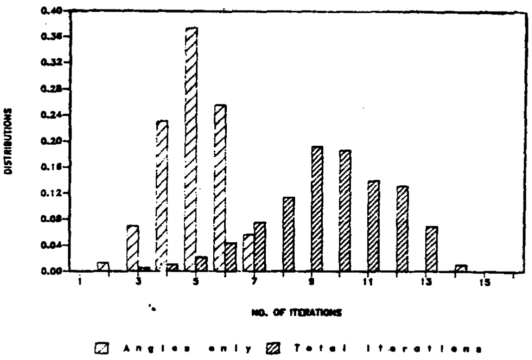
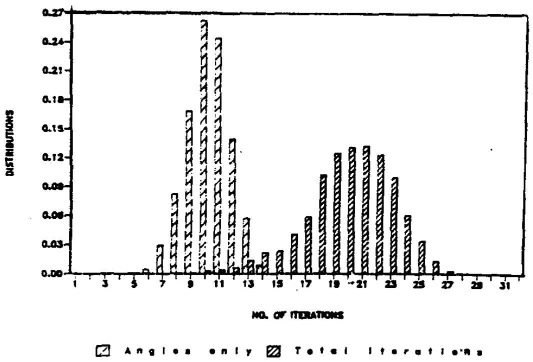


FIGURE 2. ANGLE RECODING (N = 32)



REFERENCES

[1] H.M. Ahmed, *"Signal Processing Algorithms and Architectures"*, Ph.D. dissertation, Dept. of Electrical Engineering, Stanford University, Stanford CA, June 1982.

[2] A.M. Despain, " Fourier Transform Computers using CORDIC iterations," *IEEE Transactions on Computers*, Vol. 23, pp.993-1001, Oct. 1974.

[3] Y.H. Hu, "Pipelined Cordic Architecture for the Implementation of Rotational based algorithm," *Proc., of International Conf., on VLSI tech., systems, & appl.*, Taipei, Taiwan, ROC, May 8-10, 1985.

[4] Y.H. Hu, T.Y. Sung, and H.J. Yu, "Doubly pipelined CORDIC array for digital signal processing algorithms," *Proc., of International Conf., on Circuits and Systems, San Jose, CA*, May 1986.

[5] Y.H. Hu and S. Naganathan, "A novel implementation of Chirp Z-Transform using a CORDIC processor", submitted to *IEEE Transactions on ASSP, Dec. 1988.*

[6] J.E. Volder, "The CORDIC Trigonometric computing technique," *IRE Transactions on Electronic Computers* , Vol., EC-8, No.3, pp. 330-334, Sept. 1959.

[7] J.S. Walther, "A Unified algorithm for elementary functions " *Spring Joint Computer Conference* , 1971.

[8] J.P. Hayes, *"Computer Architecture and Organization"*, McGraw Hill, 1979.