MIXED-SCALING-ROTATION CORDIC (MSR-CORDIC) ALGORITHM AND ARCHITECTURE FOR SCALING-FREE HIGH-PERFORANCE ROTATIOAL OPERATIONS

Zhi-Xiu (Phil) Lin and An-Yeu (Andy) Wu

Graduate Institute of Electronics Engineerin, and
Department of Electrical Engineering, National Taiwan University
Taipei, 106, Taiwan, R.O.C.
Contact people: Prof. An-Yeu (Andy) Wu, E-mail: andywu@cc.ee.ntu.edu.tw

Abstract

In this paper, we propose the *Mixed-Scaling-Rotation CORDIC (MSR-CORDIC)* algorithm which merges micro-rotation operation and scaling operation in conventional CORDIC algorithms to eliminate the overhead of the scaling operation. At the system architecture level, we propose the *Data-Path-Selection (DPS)* strategy for the tradeoff between hardware complexity and quantization error performance. In general, the CORDIC algorithms will suffer from the roundoff noise in fixed-wordlength implementations. We propose two schemes to control and reduce the impairment. Our simulation results show that MSR-CORDIC enhances the SQNR performance, computing speed (reducing the iteration number), and reduces the hardware complexity when compared with the newly proposed *EEAS-CORDIC* [4] Algorithm.

1. Introduction

The *COordinate Rotational Digital Computer (CORDIC)* algorithm is a well-known hardware-efficient iterative algorithm for the computation of elementary arithmetic functions such as trigonometric, hyperbolic, exponential, and logarithmic operations [1]. The CORDIC algorithm can be also applied to the rotation-based arithmetic functions, for example fast Fourier transformation (FFT), QRD-RLS filtering, EigenValue Decomposition (EVD), and Singular Value Decomposition (SVD).

In this paper, we propose a new scheme to enhance the CORDIC at both algorithmic and architectural levels. The proposed generalized MSR-CORDIC with $N = 3$ (total iteration number) and $N_{spt} = 3$ (Number of Signed of Power Two terms) design has 21.98 dB improvements compared with EEAS-CORDIC with $RT = 3$ (total iteration number) [4]. Furthermore, we can save up 33.3% hardware complexity and speed up the computation by 1.5 times than the EEAS-CORDIC,

while with better error performance.

The rest of the paper is organized as follows. We propose the MSR-CORDIC algorithm in Section 2 and show its system architecture in Section 3. The Data-Path-Selection strategy is applied to tradeoff between the hardware consumption and SQNR performance. Then, in Section 4, we make some simulations and compare system performance. Finally, Section 5 concludes our work.

2. The Proposed MSR-CORDIC Algorithm

In conventional CORDIC algorithms, the scaling factor is always greater than 1. Therefore, it is necessary to scale down the norm of the input vector to its initial value (say, unit circle), after the rotation mode is finished. Furthermore, the SQNR will be reduced due to the growth of the scaling factor. To alleviate the disadvantage of the SQNR reduction, the input vector has to keep as close as to unit circle in each iteration. Additionally, to avoid the overhead of the scaling operation, the product of the scaling factors must be equal to 1; equivalently, $\prod p_n = 1$. To overcome these problems, the range of the scaling factors must be greater and less than 1. Based on the idea, we reformulate the iterative arithmetic as

For n = 0, 1, …, N

I. Rotation phase

$$\begin{bmatrix} x(n+1) \\ y(n+1) \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{J}\mu_j 2^{-s_j} & -\sum_{i=1}^{I}\mu_i 2^{-s_i} \\ \sum_{i=1}^{I}\mu_i 2^{-s_i} & \sum_{j=1}^{J}\mu_j 2^{-s_j} \end{bmatrix} \begin{bmatrix} x(n) \\ y(n) \end{bmatrix}; \qquad (1)$$

− *Elementary angle*

$$\theta_{n+1} = \tan^{-1}\left(\frac{\sum_{i=1}^{I}\mu_i 2^{-s_i}}{\sum_{j=1}^{J}\mu_j 2^{-s_j}} \right) \qquad (2)$$

– *Accumulation angle*

$$z(n+1) = z(n) + \theta_{n+1} \qquad (3)$$

II. Scaling phase

– *Scaling factor*

$$p_{n+1} = \sqrt{\left(\sum_{i=1}^{I} 2^{-s_i}\right)^2 + \left(\sum_{j=1}^{J} 2^{-s_j}\right)^2} \qquad (4)$$

– *Product of the scaling factor*

$$\overline{p_{n+1}} = \overline{p_n} \times p_{n+1} \qquad (5)$$

End

Where $\mu_i, \mu_j \in \{-1,0,1\}$; I and J denote the number of SPT terms of $x(n)$ and $y(n)$, referred to as the *Extending Factor*. N_{spt} is the sum of I and J; θ_n is the elementary angle and the initial value. $\overline{p_{n+1}}$ denotes the product of the scaling factors in the *n*-th iteration. The initial value of $\overline{p_1}$ is 1. N denotes the total number of iteration. $s_n \in \{0,1,...,S\}$, and S denotes the number of maximum shift.

The proposed modified CORDIC algorithm is called *Mixed-Scaling-Rotation CORDIC (MSR-CORDIC)*. The reason is that we now need not to perform the micro-rotation operation and scaling operations separately. Eqs. (1-5) show that the $x(n+1)$ and $y(n+1)$ are *rotated and scaled simultaneously* in one iteration. In the conventional CORDIC and EEAS-CORDIC algorithms, the norms of both schemes are enlarged after the micro-rotation operations. On the contrary, in the proposed MSR-CORIDC algorithm, the factor P_n can be either greater or less than 1. By taking the advantage of the property of P_n, two schemes are proposed to control the dynamic range efficiently. Some other interesting features of the proposed scheme are discussed below:

1. According to Eq. (2), the angles in MSR-CORDIC is much denser than conventional CORDIC and EEAS-CORDIC, hence, the MSR-CORDIC can reach the target angle with fewer iteration. We illustrate in Fig. 1(d) by using two iterations. Furthermore, if we design the parameters, s_i, μ_i, appropriately so that both the quantization error of rotation angles and norms meet the system performance requirement at the same time; then the scaling operation can be avoid. Since we do not need the extra scaling operations, the MSR-CORDIC is faster in computational speed and the corresponding hardware cost is reduced.

2. In some applications, the rotation angles are larger than $\pi/4$, such as the twiddle factors in FFT. It is difficult for the conventional CORDIC to perform such the rotation angle. In MVR-CORDIC [2], the authors utilize the pre-rotation strategy to overcome the problem and have the improvement of error performance. However, extra hardware costs and also the
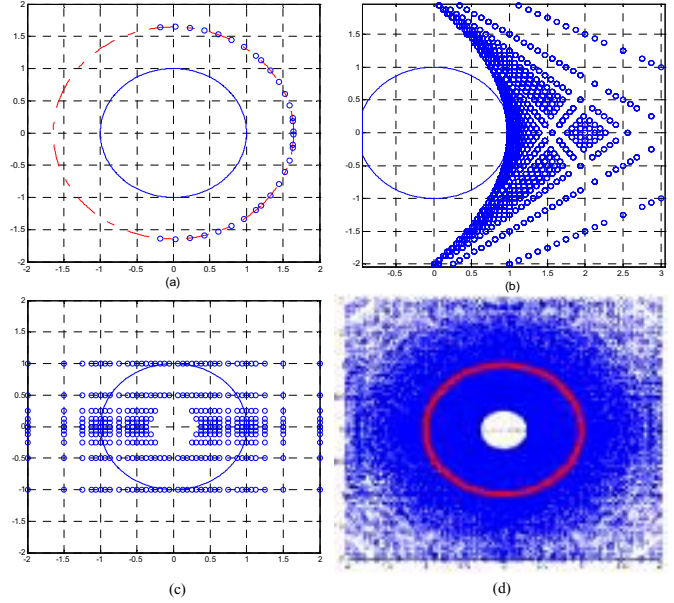

Fig. 1 Constellation of reachable points under the rotation process. (a) Conventional CORDIC with $N = R_m = 4$. (b) EEAS-CORDIC with maximum shift range $S = 4$ and $R_m = 2$. (c) MSR-CORDIC with $I = 2$, $J = 1$, and $N = 1$. (d) MSR-CORDIC with $I = 2$, $J = 1$, and $N = 2$ for $1/3 \leq P_n \leq 3$ with maximum shift range $S = 4$.

computing speed decreases. On the contrary, in the proposed MSR-CORDIC algorithm, the reachable angles are distributed from 0 to 2π.

3. VLSI Architecture of MSR-CORDIC

In this section, we will illustrate the generalized structure for the MSR-CORDIC algorithm and proposed the idea of Data-Path-Selection schemes to enhance the SQNR performance by using switches.

3.1 Normal MSR-CORDIC Structure

Firstly, we reformulate the Iteration Equations of (1) as

$$x(n+1) = \sum_{j=1}^{J} \mu_j 2^{-s_j} x(n) - \sum_{i=1}^{I} \mu_i 2^{-s_i} y(n); \qquad (6)$$

$$y(n+1) = \sum_{i=1}^{I} \mu_i 2^{-s_i} x(n) + \sum_{j=1}^{J} \mu_j 2^{-s_j} y(n). \qquad (7)$$

Both of $x(n+1)$ and $y(n+1)$ are linear combination of their prior $x(n)$ and $y(n)$. All the coefficients of $x(n)$ and $y(n)$ are power of two numbers with the sign μ_i and μ_j, respectively. Hence, two *Barrel Shifter Arrays (BSAs)* are used to perform shifting operations. The number of the output signal is N_{spt} in each BSA. To sum the outputs, $2(N_{spt}-1)$ add/subtract operations must be performed. Take $N_{spt} = 3$ for example, all cases are listed in Table 1. The architecture of Case 2 are shown in Fiq. 2(a).

TABLE 1   Four Cases in MSR-CORDIC with $N_{spt} = I + J = 3$

Case	I	J	Equation
I (*Scaling Type*)	3	0	$x(n+1) = \mu_0 2^{-s_0} x(n) + \mu_1 2^{-s_1} x(n) + \mu_2 2^{-s_2} x(n)$ $y(n+1) = \mu_0 2^{-s_0} y(n) + \mu_1 2^{-s_1} y(n) + \mu_2 2^{-s_2} y(n)$
II (*Normal type*)	2	1	$x(n+1) = \mu_0 2^{-s_0} x(n) + \mu_1 2^{-s_1} x(n) + \mu_2 2^{-s_2} y(n)$ $y(n+1) = \mu_0 2^{-s_0} y(n) + \mu_1 2^{-s_1} y(n) + \mu_2 2^{-s_2} x(n)$
III (*Normal type*)	1	2	$x(n+1) = \mu_0 2^{-s_0} x(n) + \mu_1 2^{-s_1} y(n) + \mu_2 2^{-s_2} y(n)$ $y(n+1) = \mu_0 2^{-s_0} y(n) + \mu_1 2^{-s_1} x(n) + \mu_2 2^{-s_2} x(n)$
IV (*Exchange-Scaling Type*)	0	3	$x(n+1) = \mu_0 2^{-s_0} y(n) + \mu_1 2^{-s_1} y(n) + \mu_2 2^{-s_2} y(n)$ $y(n+1) = \mu_0 2^{-s_0} x(n) + \mu_1 2^{-s_1} x(n) + \mu_2 2^{-s_2} x(n)$

3.2   Generalized MSR-CORDIC Structure

Next, we will introduce the *Data-Path-Selection* Strategy to generalize the MSR-CORDIC structure. By using the approach, the *Generalized MSR-CORDIC* structure has the better SQNR performance than any normal one.

In all the normal of MSR-CORDIC structures, all the output signals of two BSAs are fixed. If we employ the switches to control the output signals, then *x(n+1)* and *y(n+1)* can be linear combination of the shifted version of *x(n)* and *y(n)* with arbitrary *I* and *J*, which satisfies $I + J = N_{spt}$. The generalized MSR-CORDIC structures are illustrated by Fig. 2(a), in which the *Control Unit* is in charge of controlling BSAs, switches, and adders/subtractors. Three switches are used to control the output signal of two BSAs based on the signal from the controller. Fig. 2 (b) shows the complete data-paths in Case II, and Fig. 2 (c) depicts the operation of switches.

In summary, the MSR-CORDIC architecture is very regular and modular. It is very VLSI-friendly, and can be easily implemented in pipeline and parallel. For a very high computational speed requirement, each rotation operation can employ one copy of the MSR-CORDIC rotation circuit to accelerate the computing speed. For example, in a CORDIC-based orthogonal IIR digital filters design, the MSR-CORDIC is appropriately applied. Moreover, due to the fixed rotation angles, the BASs, switches, Control Unit, and ROM can be eliminated in the normal type MSR-CORDIC. Therefore, we only need few adders/subtractors to implement this design.
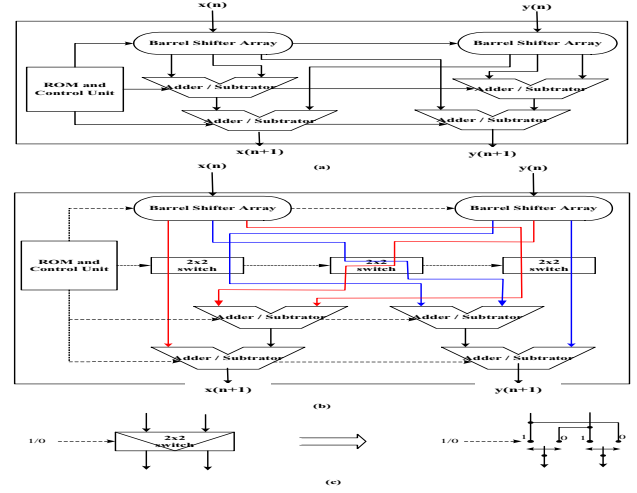

Fig. 2 (a) Normal type MSR-CORDIC with I=2, J=1. (b) Generalized MSR-CORDIC structure with $N_{spt} = 3$. Data-Path of the output signals of BSAs in case II. (c) Operation of the 2x2 switching box.

4.   Simulation Comparisons and Results

In this section, we will conduct computer simulations to show the effectiveness of the proposed schemes. The measurement of error performance is the averaged SNQR, which is obtained based on the ensemble average of 512 angles from π /2048 to π /4 with equal space. The optimal design parameters are obtained by employing exhaustive searching method for all design cases.

A.   SQNR Performance of MSR-CORDIC and EEAS-CORDIC.

The simulation is used to shows SQNR performance among two kinds of the proposed general type MSR-CORDIC and EEAS-CORDIC.

Based on averaging the SQNR from $S = 3$ to 8, the MSR-CORDIC with $N_{spt} = 3$ has 21.98 dB improvements over EEAS-CORDIC. Meanwhile, in the case of $N_{spt} = 4$, MSR-CORDIC has better SQNR improvements of 18.20 dB than EEAS-CORDIC, as shown in Fiq. 3.

B.   Hardware Cost and Computational Speed

We use the generalized MSR-CORDIC with $N_{spt} = 3$, 4 and EEAS-CORDIC to compare the hardware cost and computational speed aiming at obtaining the similar SQNR performance. The hardware cost ratio of EEAS and MSR-CORDIC is equal to the ratio of each iteration number in Case (a) and (b). Fig. 4 shows that the hardware cost of the EEAS-CODIC is 1.5 times of MSR-CODIC to achieve the similar SQNR performance. In the cases (c) and (d), the ratio must be multiplied by 1.5. The simulation results that

EEAS-CORDIC need hardware cost more 4/3 times than MSR-CORDIC. In other words, our design has less iteration number so as that the computing speed of the MSR-CORDIC is faster than EEAS-CORDIC.

C. SQNR Performance Analysis among the MSR-CORDIC Family with Nspt = 4

In this simulation, we compare the generalized and normal MSR-CORDIC with N_{spt} = 4. As expected (see Fiq. 5), the generalized MSR-CORDIC has best error performance at the penalty of the extra 4 switches. In normal MSR-CORDIC, the guidelines below will lead better SQNR performance.

a. Take both I and J are equal to N_{spt}/ 2, when N_{spt} is even.

b. Take $I = (N_{spt}+1)$ / 2, and $J = I - 1$, when N_{spt} is old.

D. Variance Comparison

We investigate the 2nd-order statistical property, the variance of SQNR, in our analysis. The variance are normalized as

$$\sigma^2{}_{SQNR} = \frac{Var(SQNR)}{mean(SQNR)} . \qquad (8)$$

The simulation result shows that the error distribution is more compact as the mean of SQNR becomes higher. Furthermore, *MSR-CORDIC algorithm is less sensitive to the kinds of MSR-CORDIC architecture with the similar SQNR performance,* illustrated by Fiq. 6.

5.   Conclusions

In this paper, we proposed a novel CORDIC scheme, MSR-CORDIC algorithm. The key idea in this algorithm is to merge two operation modes to eliminate the scaling operation (scaling-free). In practical implementation, the proposed MSR-CORDIC can be appropriately applied to various DSP systems, which require the high computational speed and angles are known in advance. In summary, based on our proposed algorithm, *Data-Path-Selection* strategy, and two searching schemes, our design can enhance SQNR performance of either 1st-order or 2nd-order statistical properties.

References

[1] J.E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electronic Comput.*, vol. EC-8, pp. 330-334, 1959.

[2] C.S Wu and A.Y. Wu, "Modified vector rotational CORDIC (MVR-CORDIC) algorithm and architecture", *IEEE Trans. Circuits and systems-II: analog and digital signal processing*, vol. 48, No. 6, June 2001.

[3] A. Y. Wu and Cheng-Shing Wu, "A Unified View for Vector Rotational CORDIC Algorithms and Architectures based on Angle Quantization Approach," to appear in *IEEE Trans. Circuits and Systems Part-I: Fundamental Theory and Applications,* Oct. 2002.

[4] C. S. Wu and A. Y. Wu, "A novel trellis-based searching scheme for EEAS-based CORDIC algorithm" In Proc. of 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 2 , pp. 1229 -1232.

[5] Y. H. Hu, "The quantization effects of the CORDIC algorithm," *IEEE Trans. Signal Processing*, vol. 40, pp. 834–844, Apr. 1992.
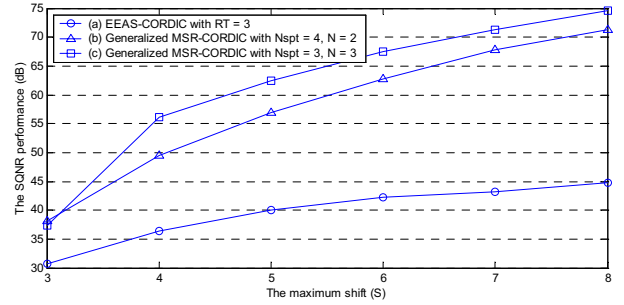
Fig. 3 The SQNR performance comparison among two types of MSR-CORDIC and EEAS-CORDIC.
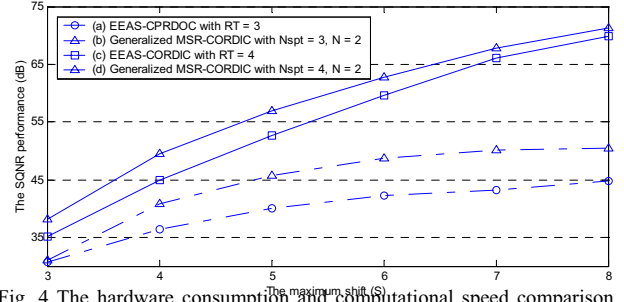

Fig. 4 The hardware consumption and computational speed comparison between the MSR-CORDIC and EEAS-CORDIC
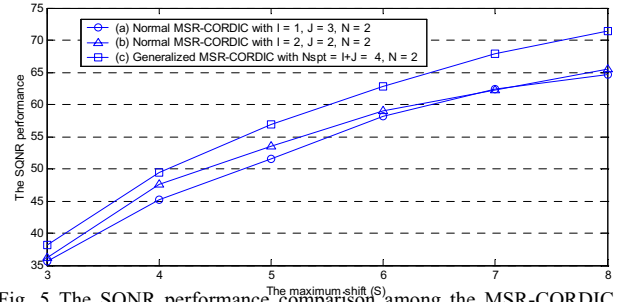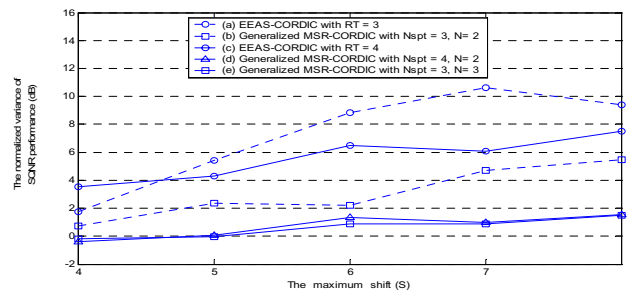

Fig. 5 The SQNR performance comparison among the MSR-CORDIC family

Fig. 6 The analysis of the SQNR variance among EEAS and MSR-CORDIC