CORDIC Designs for Fixed Angle of Rotation

Pramod Kumar Meher, *Senior Member, IEEE*, and Sang Yoon Park, *Member, IEEE*

***Abstract*—Rotation of vectors through fixed and known angles has wide applications in robotics, digital signal processing, graphics, games, and animation. But, we do not find any optimized coordinate rotation digital computer (CORDIC) design for vector-rotation through specific angles. Therefore, in this paper, we present optimization schemes and CORDIC circuits for fixed and known rotations with different levels of accuracy. For reducing the area- and time-complexities, we have proposed a hardwired pre-shifting scheme in barrel-shifters of the proposed circuits. Two dedicated CORDIC cells are proposed for the fixed-angle rotations. In one of those cells, micro-rotations and scaling are interleaved, and in the other they are implemented in two separate stages. Pipelined schemes are suggested further for cascading dedicated single-rotation units and bi-rotation CORDIC units for high-throughput and reduced latency implementations. We have obtained the optimized set of micro-rotations for fixed and known angles. The optimized scale-factors are also derived and dedicated shift-add circuits are designed to implement the scaling. The fixed-point mean-squared-error of the proposed CORDIC circuit is analyzed statistically, and strategies for reducing the error are given. We have synthesized the proposed CORDIC cells by Synopsys Design Compiler using TSMC 90-nm library, and shown that the proposed designs offer higher throughput, less latency and less area-delay product than the reference CORDIC design for fixed and known angles of rotation. We find similar results of synthesis for different Xilinx field-programmable gate-array platforms.**

***Index Terms*—Coordinate rotation digital computer (CORDIC), digital arithmetic, digital signal processing (DSP) chip, VLSI.**

I. Introduction

CORDIC stands for coordinate rotation digital computer. The key concept of CORDIC arithmetic is based on the simple and ancient principles of 2-D geometry. But the iterative formulation of a computational algorithm for its implementation was first described in 1959 by Volder [1], [2] for the computation of trigonometric functions, multiplication, and division. Not only a wide variety of applications of CORDIC have been suggested over the time, but also a lot of progress has taken place in the area of algorithm design and development of architectures for high-performance and low-cost hardware solutions [3]–[12].

Rotation of vectors through a fixed and known angle has wide applications in robotics, graphics, games, and animation [4], [13], [14]. Locomotion of robots is very often performed by successive rotations through small fixed angles and translations of the links. The translation operations are realized by simple additions of coordinate values while the new coordinates of a rotational step could be accomplished by suitable successive rotations through a small fixed angle which could be performed by a CORDIC circuit for fixed rotation [4]. Similarly, interpolation of orientations between key-frames in computer graphics and animation could be performed by fixed CORDIC rotations [14]. There are plenty of examples of uniform rotation starting from electrons inside an atom to the planets and satellites. A simple example of uniform rotations is the hands of an animated mechanical clock which perform one degree rotation each time. There are several cases where high-speed constant rotation is required in games, graphic, and animation. The objects with constant rotations are very often used in simulation, modelling, games, and animation. Efficient implementation of rotation through a known small angle to be used in these areas could be implemented efficiently by simple and dedicated CORDIC circuits. Similarly, the multiplication of complex number with a known complex constant (which is the same as the rotation of vectors through a fixed and known angle) is often encountered in communication, signal processing and many other scientific and engineering applications. In some early works, CORDIC circuits have been developed for the implementation of complex multiplications to be used for digital signal processing (DSP) applications [16]–[18], but we do not find any detailed study pertaining to efficient CORDIC realization of fixed and known-angle rotations and constant complex multiplication.

Latency of computation is the major issue with the implementation of CORDIC algorithm due to its linear-rate convergence [19]. It requires $(n+1)$ iterations to have n-bit precision of the output. Overall latency of computation increases linearly with the product of the word-length and the CORDIC iteration period. The speed of CORDIC operations is, therefore, constrained either by the precision requirement (iteration count) or the duration of the clock period. The angle recoding (AR) schemes [5]–[9] could be applied for reducing the iteration count for CORDIC implementation of constant complex multiplications by encoding the angle of rotation as a linear combination of a set of selected elementary angles of micro-rotations. In the conventional CORDIC, any given rotation angle is expressed as a linear combination of n values of elementary angles that belong to the set $\{(\sigma \cdot \arctan(2^{-r})) : \sigma \in \{-1, 1\}, r \in \{1, 2, \ldots, n-1\}\}$ to obtain an n-bit value of $\theta = \Sigma_{i=0}^{n-1}[\sigma_i \cdot \arctan(2^{-i})]$. However, in AR methods, this constraint is relaxed by adding zero into the linear combination to obtain the desired angle using relatively fewer terms of the form $(\sigma \cdot \arctan 2^{-r})$ for $\sigma \in \{1, 0, -1\}$. The elementary-angle-set (EAS) used by AR scheme is given by $S_{\mathrm{EAS}} = \{(\sigma \arctan 2^{-r}) : \sigma \in \{-1, 0, 1\}, r \in \{1, 2, \ldots, n-1\}\}$. Hu and Naganathan [5] have proposed an AR method based on the greedy algorithm that tries to represent the remaining angle using the closest elementary angle $\pm \arctan 2^{-i}$. Using this re-

Manuscript received February 15, 2011; revised December 14, 2011; accepted January 30, 2012. Date of publication March 07, 2012; date of current version January 17, 2013.

The authors are with the Institute for Infocomm Research, Agency for Science, Technology, and Research (A STAR), 138632, Singapore (e-mail: pkmeher@i2r.a-star.edu.sg; sypark@i2r.a-star.edu.sg).

Digital Object Identifier 10.1109/TVLSI.2012.2187080

coding schemes the total number of iterations could be reduced to less than half of the conventional CORDIC algorithm for the same accuracy. Wu *et al.* [7] have suggested an AR scheme based on an extended elementary-angle-set (EEAS), that provides a more flexible way of decomposing the target rotation angle. In the EEAS approach, the set S_{EAS} of the elementary-angle set is extended further to $S_{\text{EEAS}} = \{(\arctan(\sigma_1 \cdot 2^{-r_1} + \sigma_2 \cdot 2^{-r_2})) : \sigma_1, \sigma_2 \in \{-1, 0, 1\} \text{ and } r_1, r_2 \in \{1, 2, \ldots, n-1\}\}$. EEAS has better recoding efficiency in terms of the number of iterations and can yield better error performance than the AR scheme based on EAS. But the iteration period for EEAS is longer, and involves double the numbers of adders/subtractors in the CORDIC cell compared with that of the other. Most of the advantages gained in the AR schemes are cancelled out by the hardware and time involved in scaling the pseudo-rotated vector.

Since the angle of rotation for the fixed rotation case is known *a priori*, it is desirable to perform exhaustive search to obtain an optimal EAS instead of greedy search. Moreover, it is observed that the hardware-complexity of barrel-shifters alone is nearly half of that of a CORDIC circuit. We therefore aim at suggesting some techniques to minimize the complexity of barrel shifters. CORDIC computation is inherently sequential. Therefore, CORDIC is not suitable for parallel implementation, while it is a natural candidate for pipeline implementation. But, the efficient pipelined realization of CORDIC for fixed-angle vector rotations is yet to be exploited.

Keeping these in view, in this paper, we present the optimization schemes for reducing the number of micro-rotations and for reducing the complexity of barrel-shifters for fixed-angle vector-rotation. We also derive a cascaded pipelined circuit for this class of problem which is faster and involves less area-delay complexity than the existing approaches. The contributions of this paper are as follows.

1) Optimized set of micro-rotations are derived for the implementation of fixed-angle vector-rotation.
2) Shift-add operations for corresponding scaling circuits are derived.
3) A novel hardware pre-shifting scheme is suggested for reduction of barrel-shifter complexity.
4) Single-rotation and bi-rotation CORDIC circuits are designed and used to derive cascaded CORDIC for high-speed fixed-angle vector rotations.
5) The fixed-point mean-squared-error (MSE) of the proposed CORDIC circuit is analyzed, and an efficient strategy for reducing the error is described.

The remainder of this paper is organized as follows. Section II deals with the optimization of elementary angle set for different accuracies of implementation. Efficient circuits for implementation of micro-rotations for fixed rotations are presented in Section III. Implementation of scaling is discussed in Section IV. Section V analyzes the MSE of the proposed CORDIC. Hardware and time complexities are given and synthesis results of the proposed designs are compared with the conventional and a reference design in Section VI. Conclusions are presented in Section VII.

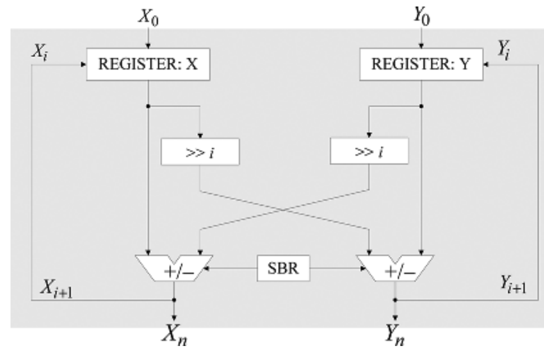

Fig. 1. Reference CORDIC circuit for fixed rotations.

II. Optimization of Elementary Angle Set

The rotation-mode CORDIC algorithm to rotate a vector $\mathbf{U} = [U_x \; U_y]^T$ through an angle ϕ to obtain a rotated vector $\mathbf{V} = [V_x \; V_y]^T$ is given by [1], [2]

$$(U_x)_{i+1} = (U_x)_i - \sigma_i \cdot (U_y)_i \cdot 2^{-i} \tag{1a}$$
$$(U_y)_{i+1} = (U_y)_i + \sigma_i \cdot (U_x)_i \cdot 2^{-i} \tag{1b}$$
$$\phi_{i+1} = \phi_i - \sigma_i \tan^{-1}(2^{-i}) \tag{1c}$$

such that when n is sufficiently large

$$\begin{bmatrix} V_x \\ V_y \\ \phi \end{bmatrix} \leftarrow T \begin{bmatrix} (U_x)_n \\ (U_y)_n \\ 0 \end{bmatrix} \tag{1d}$$

where $\sigma_i = -1$ if $\phi_i < 0$ and $\sigma_i = 1$ otherwise, and T is the scale-factor of the CORDIC algorithm, given by

$$T = \prod_{i=0}^{n-1} \left[1 + 2^{-2i}\right]^{-1/2}. \tag{2}$$

In case of fixed rotation, ϕ_i could be pre-computed and the sign-bits corresponding to σ_i could be stored in a sign-bit register (SBR) in CORDIC circuit. The CORDIC circuit therefore need not compute the remaining angle ϕ_i during the CORDIC iterations [3].

A reference CORDIC circuit for fixed rotations according to (1a) and (1b) is shown in Fig. 1. X_0 and Y_0 are fed as set/reset input to the pair of input registers and the successive feedback values X_i and Y_i at the ith iteration are fed in parallel to the input registers. Note that conventionally we feed the pair of input registers with the initial values X_0 and Y_0 as well as the feedback values X_i and Y_i through a pair of multiplexers.

We show here that for rotation of a vector through a known and fixed angle of rotation using a rotation-mode CORDIC circuit, we can find a set of a small number of predetermined elementary angles $\{\alpha_i, \text{ for } 0 \le i \le m-1\}$, where $\alpha_i = \arctan(2^{-k(i)})$ is the elementary angle to be used for the ith micro-rotation in the CORDIC algorithm (1), and m is the minimum necessary number of micro-rotations. Meanwhile, it is well known that the rotation through any angle, $0 < \theta \le 2\pi$ can be mapped into a positive rotation through $0 < \phi \le \pi/4$ without any extra arithmetic operations [10]. Hence, as a first step of optimization, we perform the rotation mapping so that the rotation angle lies in the range of $0 < \phi \le \pi/4$. In the next step, we minimize the number of elementary angles in the set $\{\alpha_i\}$ according to the accuracy requirements. The rotation mode CORDIC algorithm of (1), therefore, can be modified accordingly to have

$$\begin{bmatrix}(U_x)_{i+1}\\(U_y)_{i+1}\end{bmatrix} = \begin{bmatrix}1 & -\sigma_i 2^{-k(i)}\\ \sigma_i 2^{-k(i)} & 1\end{bmatrix}\begin{bmatrix}(U_x)_i\\(U_y)_i\end{bmatrix} \quad \text{(3a)}$$

such that for a minimum number m

$$\begin{bmatrix}U'_x\\U'_y\\\phi_A\end{bmatrix} \leftarrow K\begin{bmatrix}(U_x)_m\\(U_y)_m\\0\end{bmatrix}. \quad \text{(3b)}$$

The scale-factor K now depends on the set $\{\alpha_i\}$. The accuracy of CORDIC algorithm depends on how closely the resultant rotation ϕ_A due to all the micro-rotations in (1) approximates to the desired rotation angle ϕ, which in turn determines the deviation of actual rotation vector from the estimated value. We show here that only a few elementary angles are sufficient to have a CORDIC rotation in the range $[0,\ \pi/4]$, and different sets of elementary angles can be chosen according to the accuracy requirement.

The simple pseudo code to optimize a set of micro-rotations is described in **Algorithm 1**. If the maximum accuracy ϵ_ϕ which is defined as the maximum tolerable error between desired angle and approximated angle is given as an input, the optimization algorithm searches the parameters $k(i)$ and σ_i that can minimize an objective function $\Delta\phi$. The algorithm starts with the single micro-rotation, i.e., $m = 1$, then if the micro-rotation that has smaller angle of deviation than ϵ_ϕ cannot be found, the number of micro-rotations is increased by one and the optimization algorithm is run again. Exhaustive search is employed in the optimization algorithm to search the entire parameter space for all the combinations of $k(i)$ and σ_i. Based on the obtained micro-rotations, the parameters for scaling operation can be searched with the different objective function, which is described in Section IV. The sub-optimal set of micro-rotations may be used in some cases, if the optimal set of micro-rotations cannot satisfy the design constraint for scaling. We have used sub-optimal solutions particularly for the rotation with the angle of 31° and 35° in Table I since the scaling requires more terms in these two cases if optimal solutions are used.

Algorithm 1 Obtains the Optimal Micro-Rotations.

$m := 1$
do
  $\Delta\phi := \min|\phi - \sum_{i=0}^{m-1} \arctan \sigma_i 2^{-k(i)}|, \forall\, \sigma_i \in \pm 1,\ k(i)$ is nonnegative integer.
    $m := m + 1$
while $(\Delta\phi > \epsilon_\phi)$
  end while

In the experiment with the maximum input angular deviation $\epsilon_\phi = 0.04°$, we found that a set of four selected micro-rotations is enough. In Table I, it is shown that rotations through any angle in the range $0 < \phi \le 45°$ (in odd integer degrees) could be achieved with maximum angular deviation $\Delta\phi = 0.037°$ (0.646×10^{-3} radian), where $\Delta\phi = |\phi - \phi_A|$. Using a maximum of two selected micro-rotations, the rotations could be achieved with maximum angular deviation with $\Delta\phi = 1.875°$ (0.033 radian). In case of six micro-rotations, angular deviation $\Delta\phi$ could be reduced to $\sim 0.0005°$.

In Table II, it is shown further that rotations through $0.1° \le |\phi| \le 2.0°$ in an interval of 0.1° could be obtained by four micro-rotations with angular deviation, $\sim 0.003°$. Here we can make an observation that we can always achieve higher accuracy with more number of micro-rotations. From Table II, we find that higher accuracy could be achieved in case of small rotation angles like 1° or 2°, compared to the most of the larger angles when the same number of micro-rotations is used.

TABLE I
OPTIMIZATION OF FULL ROTATIONS WITH FOUR MICRO-ROTATIONS

$\phi°$	$k(0), s_0$	$k(1), s_1$	$k(2), s_2$	$k(3), s_3$	$\Delta\phi$
45	0, 1	−−	−−	−−	0.000
43	0, 1	5, 0	8, 0	−−	0.014
41	0, 1	4, 0	7, 0	−−	0.024
39	0, 1	3, 0	6, 1	8, 1	0.006
37	0, 1	3, 0	6, 0	−−	0.020
35	2, 1	2, 1	2, 1	3, 0	0.016
33	1, 1	3, 1	7, 0	8, 0	0.019
31	1, 1	4, 1	6, 1	−−	0.037
29	2, 1	2, 1	6, 1	−−	0.032
27	1, 1	7, 1	−−	−−	0.013
25	1, 1	5, 0	8, 1	−−	0.001
23	1, 1	4, 0	−−	−−	0.011
21	2, 1	2, 1	3, 0	10, 1	0.003
19	1, 1	3, 0	7, 0	−−	0.008
17	1, 1	2, 0	4, 1	6, 1	0.000
15	2, 1	6, 1	10, 1	−−	0.013
13	1, 1	2, 0	7, 1	−−	0.024
11	3, 1	4, 1	8, 1	10, 1	0.019
9	3, 1	5, 1	9, 1	−−	0.027
7	3, 1	9, 0	−−	−−	0.013
5	3, 1	5, 0	7, 0	9, 1	0.001
3	4, 1	7, 0	9, 0	−−	0.017
1	6, 1	9, 1	−−	−−	0.007

s_i is the sign-bit corresponding to the sign term σ_i, such that $s_i = 1$ and 0 for $\sigma_i = 1$ and -1, respectively. $\Delta\phi = |\phi - \phi_A|$.

TABLE II
OPTIMIZATION OF SMALL ROTATIONS WITH FOUR MICRO-ROTATIONS

$\phi°$	$k(0), s_0$	$k(1), s_1$	$k(2), s_2$	$k(3), s_3$	$\Delta\phi$
2.0	5, 1	8, 1	12, 0	−−	0.0003
1.9	5, 1	9, 1	−−	−−	0.0018
1.8	5, 1	13, 1	−−	−−	0.0031
1.7	5, 1	9, 0	12, 1	13, 1	0.0010
1.6	5, 1	8, 0	11, 1	13, 1	0.0011
1.5	1, 0	2, 1	2, 1	13, 0	0.0004
1.4	6, 1	7, 1	10, 1	−−	0.0013
1.3	5, 1	7, 0	11, 0	12, 0	0.0003
1.2	5, 1	7, 0	9, 0	11, 0	0.0024
1.1	6, 1	8, 1	12, 0	14, 0	0.0015
1.0	6, 1	9, 1	13, 0	−−	0.0001
0.9	6, 1	14, 1	−−	−−	0.0013
0.8	6, 1	9, 0	12, 1	−−	0.0027
0.7	7, 1	8, 1	11, 1	−−	0.0006
0.6	6, 1	8, 1	10, 0	12, 0	0.0014
0.5	7, 1	10, 1	−−	−−	0.0036
0.4	7, 1	10, 0	13, 1	−−	0.0013
0.3	7, 1	9, 0	11, 0	13, 0	0.0007
0.2	8, 1	11, 0	14, 1	−−	0.0007
0.1	9, 1	12, 0	−−	−−	0.0021

III. IMPLEMENTATION OF MICRO-ROTATIONS

Since the elementary angles and direction of micro-rotations are predetermined for the given angle of rotation, the angle es-

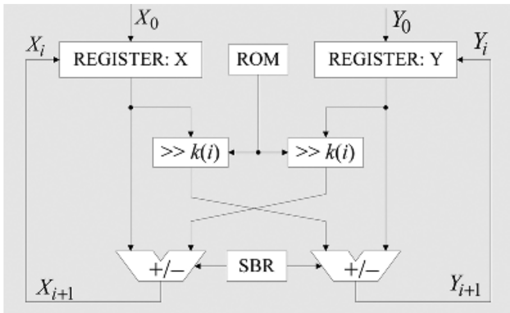

Fig. 2. CORDIC cell for constant complex multiplications.

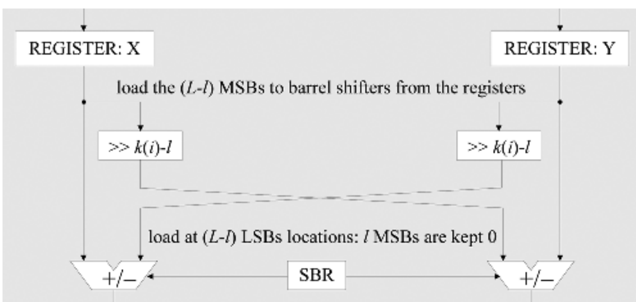

Fig. 3. Hardwired pre-shifting in basic CORDIC module.

timation data-path is not required in the CORDIC circuit for fixed and known rotations. Moreover, because only a few elementary angles are involved in this case, the corresponding control-bits could be stored in a ROM of few words. A CORDIC circuit for complex constant multiplications is shown in Fig. 2. The ROM contains the control-bits for the number of shifts corresponding the micro-rotations to be implemented by the barrel-shifter and the directions of micro-rotations are stored in the sign-bit register (SBR). The major contributors to the hardware-complexity in the implementation of a CORDIC circuit are the barrel-shifters and the adders. There are several options for the implementation of adders [22], from which a designer can always choose depending on the constraints and requirements of the application. But, we have some scope to develop techniques for reducing the complexity of barrel-shifters over the conventional designs as discussed in the followings.

1) Minimization of Barrel-Shifter Complexity by Hardwired Pre-Shifting: A barrel-shifter for maximum of S shifts for word-length L is implemented by $\lceil \log_2(S+1) \rceil$-stages of demultiplexors, where each stage requires L number of 1:2 line MUXes. The hardware-complexity of barrel-shifter, therefore, increases linearly with the word-length and logarithmically with the maximum number of shifts. We can reduce the effective word-length in the MUXes of the barrel-shifters, and so also the number of stages of MUXes by simple hardwired pre-shifting as shown in Fig. 3. If l is the minimum number of shifts in the set of selected micro-rotations, we can load only the $(L-l)$ more-significant bits (MSBs) of an input word from the registers to the barrel-shifters, since the l less significant bits (LSBs) would get truncated during shifting. The barrel-shifter, therefore, needs to implement a maximum of $(s-l)$ shifts only, where s is the maximum number of shifts in the set of selected micro-rotations. The output of the barrel-shifters are loaded as the $(L-l)$ LSBs to the add/subtract units, and the l MSBs of the corresponding operand of add/subtract unit are hardwired to 0. Therefore, the hardware-complexity of a barrel-shifter could be reduced by the hardwired pre-shifting approach. The time involved in a barrel-shifter could also be reduced by hardwired pre-shifting, since the delay of the barrel-shifter is proportional to the number of stages of MUXes, and it also be possible to reduce the number stages by hardwired pre-shifting.

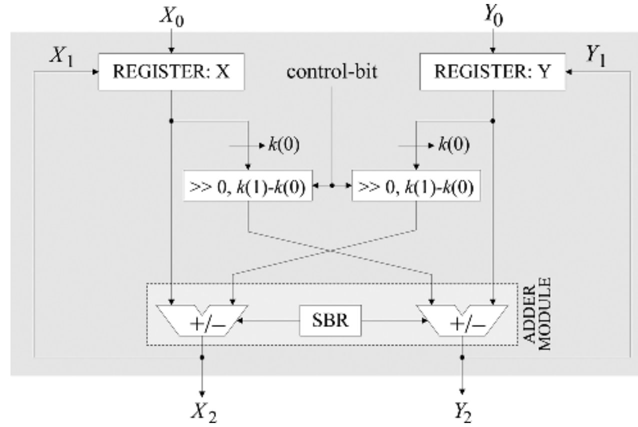

Fig. 4. Hardwired pre-shifted bi-rotation CORDIC circuit. SBR is sign-bit register of 2-bits size. $\longrightarrow k(0)$ indicates right-shift by $k(0)$ bit-locations.

In Table I, we find that the minimum number of shifts l is greater than one in more than 75% of the cases. Similarly, in Table II, we find that l is always greater than 5 except the angle $1.5°$. Using hardwired pre-shifting, it would therefore be possible to considerably reduce the total number of shifts to be implemented by barrel-shifters, so as to substantially reduce the hardware-complexity and delay of the barrel-shifters. A conventional barrel-shifter for maximum of S shifts is implemented by $\lceil \log_2(S+1) \rceil$-stages of 2:1 MUXes. But, when the number of shifts is known *a priori*, one can design the barrel-shifter to include the specific shifts. For implementing four discrete shifts (see Table I) irrespective of the maximum number of shifts, the barrel-shifter would require three stages of 2:1 MUXes by hardwiring the shifts.

2) Bi-Rotation CORDIC Cell: We find that using only two micro-rotations, it is possible to get an accuracy up to 0.033 radian. Although the accuracy achieved by two micro-rotations is inadequate in many situations, but can be used for some applications where the outputs are quantized, e.g., in case of speech and image compression, etc., [23], [24]. Besides, the rotations with four and six micro-rotations can also be implemented successively by two and three pairs of micro-rotations, respectively. Therefore, we design an efficient CORDIC circuit to implement a pair of micro-rotations, and named as "*bi-rotation* CORDIC". The proposed circuit for bi-rotation CORDIC is shown in Fig. 4. It consists of an adder-module, two 2:1 multiplexers and a sign-bit register (SBR) of two bit size. The adder-module consists of a pair of adders/subtractors. The adders/subtractors perform additions or subtractions according to the sign-bit available from the SBR. The components of the input vector (real and the imaginary parts of the input complex operand) are loaded to the input-registers through set/reset input. The output of the registers are sent in two lines where the content of the register is fed to one of the adders/subtractors directly while that in the other line is loaded

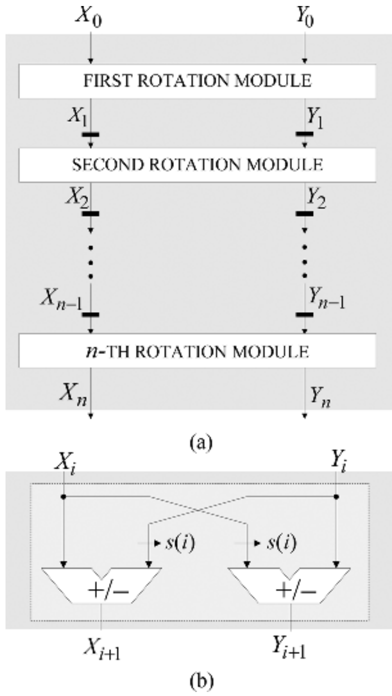

Fig. 5. (a) Multi-stage single-rotation cascaded CORDIC circuit. (b) Structure of ith rotation module. $\longrightarrow s(i)$ indicates right-shift by i bit-locations.

to the barrel-shifter pre-shifted by $k(0)$ bit-locations to right by hardwired pre-shifting technique. The output of the adders are loaded back to the input registers for the second CORDIC iteration. The bi-rotation CORDIC involves only a pair of barrel-shifters consisting of only one stage of 2:1 MUXes. The control-bit for the barrel-shifters is 0 for the first micro-rotation (no shift) and 1 for the second micro-rotation (shift through $k(1) - k(0)$). The control bits are generated by a T flip-flop, since they are 1 and 0 in each alternate cycle.

3) High-Throughput Implementation Using Cascaded Multi-Stage CORDIC: For the implementation of small rotations (the remaining angle after the first two micro-rotations), as shown in Table II, $l \geq 9$ except the angle 1.5°. Similarly, in Table I, we can notice that the second half of the micro-rotations has the minimum shifts $l \geq 5$. It would be possible to take the best advantage of hardwired-pre-shifting, if the micro-rotations are implemented in more than one CORDIC modules in separate stages in a cascade. Moreover, since the cascade of CORDIC modules is inherently pipelined, it would provide high-throughput pipelined implementation. To implement the CORDIC rotations with higher accuracy without affecting the throughput of computation, we can therefore have cascaded-multi-stage CORDIC consisting of single-rotation cells and bi-rotation CORDIC as described in the followings.

Cascaded CORDIC with Single-Rotation Cells: A multi-stage-cascaded pipelined-CORDIC circuit consisting of single-rotation modules is shown in Fig. 5. Each stage of the cascaded design consists of a dedicated rotation-module that performs a specific micro-rotation. The structure and function of a rotation-module is depicted in Fig. 5(b). Each rotation-module consists of a pair of adders or subtractors (depending on the direction of micro-rotation which it is required to implement). Each of the adders/subtractors loads one of the pair of inputs directly and loads the other input in a pre-shifted form at $(L - s(i))$ LSB locations, where $s(i)$ is the number of right-shifts required to be performed to implement the ith micro-rotation. The $s(i)$ MSB locations are hardwired to be zero. The rotation-module in this case does not require input from SBR since each adder module always performs either addition or subtraction. It also does not require barrel-shifter since it has to implement only one fixed micro-rotation. The output of each stage is latched to the input of its succeeding stage as shown in the figure. The critical-path in this case amounts to only one addition/subtraction operation in the adder module. Total latency of n-stage single-rotation cascade amounts to $n(T_A + T_{FF})$, where T_A and T_{FF}, are the addition/subtraction time and D flip-flop delay, respectively.

We find that in more than two-third of the rotation angles as shown in Table I, only three micro-rotations are adequate to have the maximum deviation of ϕ up to 0.04°. The complex multiplications involving three such micro-rotations could be implemented by three-stage-cascaded CORDIC circuit shown in Fig. 5 (for $n = 3$). The rotation using 4 and 6 micro-rotations, similarly, would require 4 and 6 stages of rotation module for pipelined implementation. This can also be implemented in non-pipelined form using $(n - 1)$ carry-propagate adders with total latency of $T_A + (\Sigma_{i=1}^{n-1} k(i)) \times T_{FA}$, where T_A and T_{FA}, are, respectively, the time required for L-bit addition-time and full-adder delay, L being word-length of implementation. $k(i)$ is the number of shifts of the ith stages.

Cascaded CORDIC with Bi-Rotation Cells: For reduction of adder complexity over the cascaded single-rotation CORDIC, the micro-rotations could be implemented by a cascaded bi-rotation CORDIC circuit. A two-stage cascaded bi-rotation CORDIC is shown in Fig. 6. The first two of the micro-rotations as shown in Table I out of the four-optimized micro-rotations could be implemented by stage-1, while the rest two are performed by stage-2. The structure and function of the bi-rotation CORDIC is shown in Fig. 4. For implementing six selected micro-rotations, we can use a three-stage-cascade of bi-rotation CORDIC cells. The three-stage bi-rotation cells could however be extended further when higher accuracy is required.

IV. Scaling Optimization and Implementation

We discuss here the optimization of scaling to match with the optimized set of elementary angles for the micro-rotations.

A. Scaling Approximation for Fixed Rotations

The generalized expression for the scale-factor given by (2) can be expressed explicitly for the selected set of m_1 micro-rotations as

$$K = \prod_{i=0}^{m_1-1} \left[(1 + 2^{-2k(i)})\right]^{-1/2} \tag{4}$$

where $k(i)$ for $0 \leq i < m_1$ is the number of shifts in the ith micro-rotation. Except for $k(i) = 0$ (i.e., rotation by 45°), by binomial expansion, any term in (4) can be written as

$$1 - \frac{x}{2} + \frac{3x^2}{8} - \frac{5x^3}{16} + \frac{35x^4}{128} - \frac{63x^5}{256} + \frac{231x^6}{1024} - \cdots \tag{5}$$

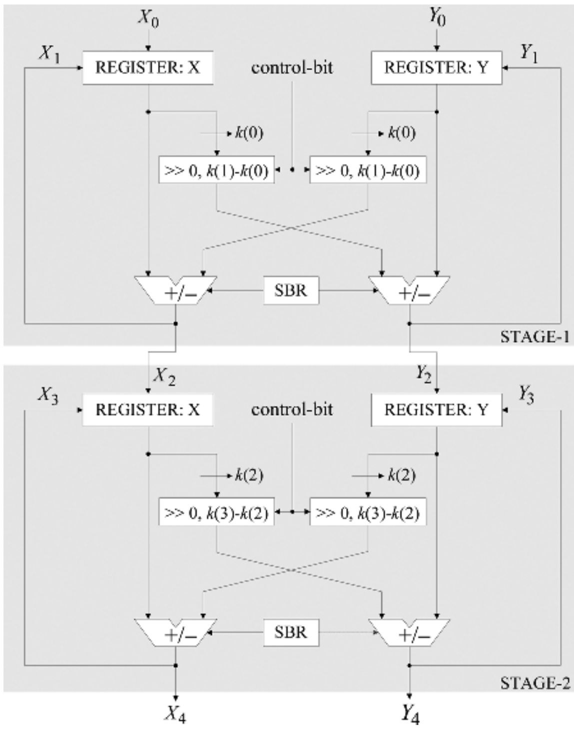

Fig. 6. Two-stage cascaded bi-rotation CORDIC circuit. SBR is sign-bit register of 2-bits size.

where $x = 2^{-2i}$, i being the number of shifts in a micro-rotation, and can be expressed alternatively in terms of i as

$$1-\frac{1}{2^{2i+1}}+\frac{3}{2^{4i+3}}-\frac{5}{2^{6i+4}}+\frac{35}{2^{8i+7}}-\frac{63}{2^{10i+8}}+\frac{231}{2^{12i+10}}-\cdots. \quad (6)$$

Replacing each term in (4) by the expression of (6), we can obtain an approximate scale-factor as a product of shift-add terms of form

$$K_A = \prod_{i=0}^{m_2-1} \left[1 + \delta_i 2^{-s(i)}\right] \quad (7)$$

where $s(i)$ is the number of shifts performed for the ith iteration of scaling, $\delta_i = \pm 1$, and m_2 is maximum number of scaling iterations required for the approximation.

The number of terms of (6), those are required to be accounted for to obtain the approximate scale-factor K_A [given by (7)] can be estimated according to value of i and the desired output accuracy which is limited by the number of micro-rotations used for the pseudorotation. The number of shifts-add/subtract terms in the expression of (7) is therefore minimized separately for the CORDIC implementations by four micro-rotations and six micro-rotations for different angles of rotation. It can be found that for four micro-rotation CORDIC implementation, where the error in ϕ is $\sim 0.04°$, only the first two terms in (6) contribute for $(0 \le i \le 4)$, while up to the third and the fifth terms contribute for $(0 \le i \le 2)$ and $(0 \le i \le 1)$, respectively. Similarly, for six micro-rotation CORDIC implementation, where the error in ϕ is $\sim 0.0005°$, the first two terms in (6) contribute for $(0 \le i \le 8)$, while up to the third, fourth and fifth terms contribute for $(0 \le i \le 3)$, $(0 \le i \le 2)$, and $(0 \le i \le 1)$, respectively. Accordingly, we have obtained the recursive shift-add expressions of scale-factor K_A in the form of (7).

TABLE III
OPTIMIZED SHIFTS TO IMPLEMENT SCALING FOR THE CASE OF ROTATION WITH FOUR MICRO-ROTATIONS

$\phi°$	$s(0), t_0$	$s(1), t_1$	$s(2), t_2$	K	K_A	ΔK
41	2, 0	4, 0	8, 1	0.7057	0.7059	0.2315
39	2, 0	4, 0	9, 0	0.7016	0.7018	0.2798
37	2, 0	4, 0	9, 0	0.7016	0.7018	0.2721
35	3, 0	5, 1	8, 1	0.9060	0.9059	0.1721
33	3, 0	6, 1	10, 0	0.8875	0.8878	0.3578
31	4, 0	4, 0	6, 1	0.8926	0.8926	0.0703
29	4, 0	8, 1	−−	0.9411	0.9412	0.1068
27	4, 0	5, 0	6, 0	0.8944	0.8940	0.4332
25	4, 0	5, 0	6, 0	0.8940	0.8940	0.0319
23	4, 0	4, 0	6, 1	0.8927	0.8926	0.0518
21	4, 0	8, 0	−−	0.9339	0.9338	0.0752
19	3, 0	6, 1	10, 0	0.8875	0.8878	0.3502
17	3, 0	7, 0	9, 0	0.8659	0.8665	0.6261
15	5, 0	10, 1	−−	0.9700	0.9697	0.3377
13	3, 0	7, 0	−−	0.8677	0.8682	0.5402
11	7, 0	9, 0	−−	0.9903	0.9903	0.0887
9	7, 0	−−	−−	0.9918	0.9922	0.3989
7	7, 0	−−	−−	0.9923	0.9922	0.0892
5	7, 0	−−	−−	0.9918	0.9922	0.4295
3	9, 0	−−	−−	0.9980	0.9980	0.0267
1	13, 0	−−	−−	0.9999	0.9999	0.0019

K is the required scale factor and K_A is the approximated scale factor. $s(i)$ is the number of shifts, t_i is the sign-bit corresponding to the sign term δ_i, such that $t_i = 1$ and 0 for $\delta_i = 1$ and -1, respectively. $\Delta K = |1 - K_A/K| \times 10^3$.

Algorithm 2 describes the optimization scheme to search the parameters $k(i)$ and σ_i. Once the set of micro-rotations is obtained by **Algorithm 1**, the ideal scaling factor K can be calculated using (4). The objective function ΔK is defined as deviation of K_A/K from 1, i.e., $\Delta K = |1 - K_A/K|$. The algorithm starts with the single term of scaling, then the number of scaling terms is increased by one until ΔK is smaller than the given maximum deviation ϵ_K, which needs to be set as the same value as ϵ_ϕ in the **Algorithm 1** since ΔK and $\Delta\phi$ contribute equally to the overall approximation error. In the experiment, we need three terms in the expression of (7) as listed in Table III in the range of 1° to 41° when ϵ_K is set as 0.698×10^{-3} which is the same value as ϵ_ϕ used to obtain the Table I after conversion of 0.04° to radian.

Algorithm 2 Obtains the Optimal Shifts for Scaling.

$K := \prod_{i=0}^{m_1-1} \left[(1 + 2^{-2k(i)})\right]^{-1/2}$
$m_2 := 1$
do
  $\Delta K := \min |1 - \prod_{i=0}^{m_2-1}[1 + \delta_i 2^{-s(i)}]/K|$, $\forall \delta_i \in \pm 1$, $s(i)$ is nonnegative integer.
  $m_2 := m_2 + 1$
while $(\Delta K > \epsilon_K)$
end while

We derive here the expression of scale factors separately for 43° and 45° rotations to get scaling with desired accuracy with less number of iterations compared with the above approach.

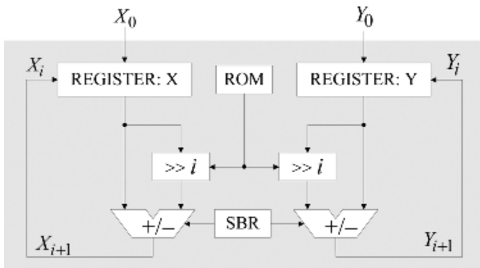

Fig. 7. Shift-add scaling circuit using hardwired pre-shifted loading.

To have an accuracy up to 0.698×10^{-3}, the scale-factor for rotation through 43° and 45° can be expressed as

$$K \simeq 1 - \frac{1}{2^2} - \frac{1}{2^4} + \frac{1}{2^6} + \frac{1}{2^8}. \quad (8)$$

Equation (8) can be expressed in recursive shift-add forms

$$K = 1 - \frac{1}{2^2}\left(1 + \frac{1}{2^2}\right)\left(1 - \frac{1}{2^4}\right). \quad (9)$$

B. Implementation of Scaling

Scaling and micro-rotations could be implemented either in the same circuit in interleaved manner or in two separate stages. The implementation of scaling as well as the micro-rotation would however depend on the level of desired accuracy, and the implementation of scaling also depends on the implementation of micro-rotations. Therefore, we discuss here the realization of the scaling circuits corresponding to different implementations of micro-rotations.

1) Generalized Implementation of Scaling: The shift-add circuit for scaling according to (7) is shown in Fig. 7. The scaling circuit of Fig. 7 can use hardwired pre-shifting for minimizing barrel-shifter complexity and could be placed after the CORDIC cell of Fig. 2 to perform micro-rotation and scaling in two separate stages. The generalized CORDIC circuit for fixed rotation to perform the micro-rotation and the scaling in interleaved manner in alternate cycles is shown in Fig. 8. The circuit of Fig. 8 is similar to that of Fig. 2. It involves only an additional line-changer circuit to change the path of unshifted (direct) input. The structure and function of line-changer is shown in Fig. 8(b). The line-changer is placed on the unshifted input data line to keep the critical path the same as that of Fig. 2.

2) Implementation of Scaling for Bi-Rotation CORDIC: The scaling and micro-rotations for the proposed bi-rotation CORDIC could be implemented in two separate pipelined stages, where the pair of micro-rotations are implemented by the CORDIC circuit of Fig. 4, and scaling is implemented by a shift-add circuit. The scale factor for this case can be represented by two shift-add terms as

$$K_A = \left(1 + \delta_0 2^{-s(0)}\right) \times \left(1 + \delta_1 2^{-s(1)}\right). \quad (10)$$

The two-factor scaling of (10) can be implemented by the shift-add circuit of Fig. 9. It consists of a pair of adders/subtractors and a pair of single-stage barrel-shifters. Each barrel-shifter consists of only one stage of 2:1 MUXes. The input of each of the barrel-shifters is hardwired pre-shifted by $s(0)$ locations to right. Each of the barrel-shifters shifts the input through $[s(1) - s(0)]$ locations to right, when the control-bit is 1. No additional shifts are required when control-bit is 0. The control-bit can be generated by a T flip-flop since it toggles in each cycle. The add-subtract cell performs addition if $t_i = 1$ and performs subtraction if $t_i = 0$, which could be controlled through a two-bit SBR.

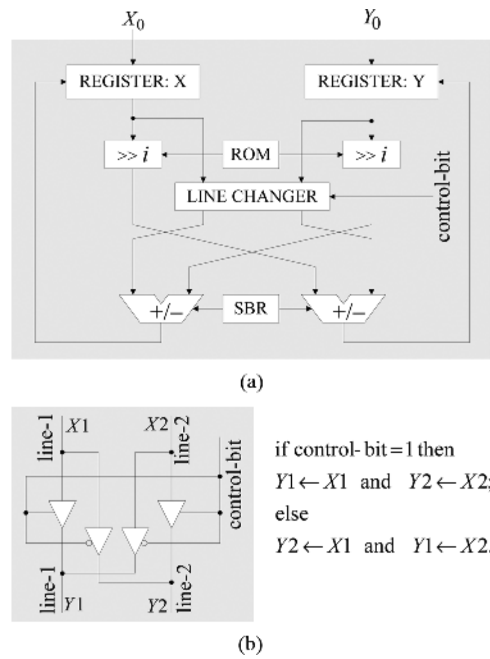

Fig. 8. CORDIC circuit for interleaved implementation of micro-rotations as well as scaling circuit. (a) The CORDIC circuit. (b) Structure and function of line-changer. For control-bit $= 1$ it performs micro-rotations and for control-bit $= 0$ it performs the shift-add operations for scaling.

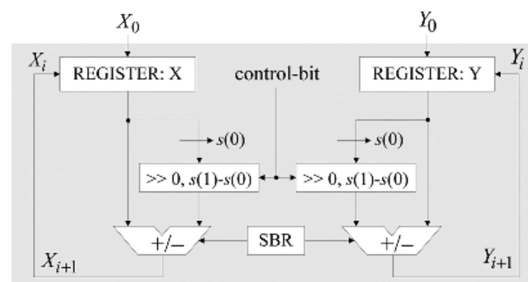

Fig. 9. Shift-add circuit of two-factor scaling using hardwired pre-shifting.

3) Implementation of Scaling for Cascaded Single-Rotation CORDIC: The shift-add circuit for single-rotation-cascaded CORDIC is shown in Fig. 10. It consists of a pair of dedicated adders-subtractors. It does not require any multiplexer or sign-bit register. A pair of input is fed to the adder/subtractor from the register, where one of the inputs is obtained directly from the content of the registers, while the other input is shifted by $s(i)$ locations to right before being fed to the adder/subtractor. The choice of adder or subtractor depends on the sign-factor in the shift-add term to be implemented by the circuit.

4) Implementation of Scaling for Cascaded Bi-Rotation CORDIC: The cascaded bi-rotation CORDIC could either be used for implementing in two or three stages for four and six micro-rotations, respectively. For scaling by three shift-add-factors as shown in Table III, we can use one

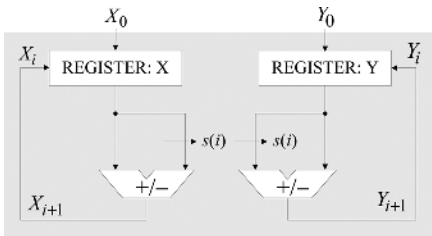

Fig. 10. Shift-add circuit for single-rotation-cascaded-scaling using hardwired pre-shifting.

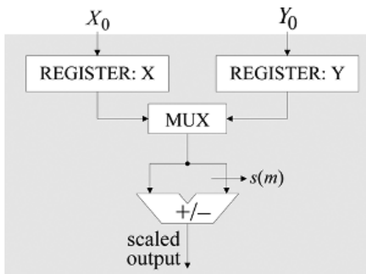

Fig. 11. Time-multiplexed shift-add circuit for one-factor scaling.

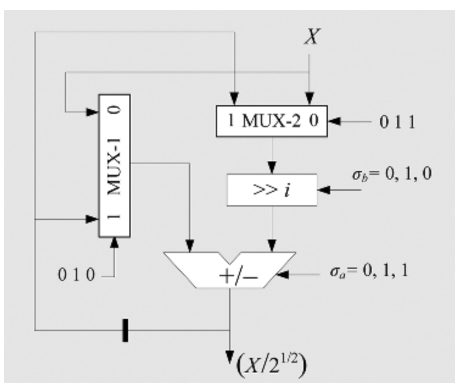

Fig. 12. Scaling circuit for 43° and 45° rotation. (a) $\sigma_a = 0$ and 1correspond to addition and subtraction, respectively. $\sigma_b = 0$ and 1correspond to two right-shifts and four right-shifts, respectively, in the barrel-shifter.

two-factor-scaling circuit of Fig. 9 and the third scaling factor could be implemented by a multiplexed shift-add circuit of Fig. 11. The scaling for six micro-rations, which involves five shift-add factors, could be implemented by a pair of two-factor scaling circuit and a multiplexed circuit.

5) Implementation of Scaling for Large Rotations: The scaling circuit for rotation through 43° and 45° based on (9) is shown in Fig. 12. We can implement this scaling also by simple modifications of cascaded forms of single-factor scaling circuit, two-factor scaling circuits and time-multiplexed scaling circuits of Figs. 9–11.

V. Analysis of Error

There are two types of error encountered during the rotation mode CORDIC iterations. Those are: approximation error and round-off error. Approximation error arises due to approximation of angle of rotation and scaling factor, while the round-off error arises due to the finite word-length of the output components. We derive the expression for these two errors in the following subsections.

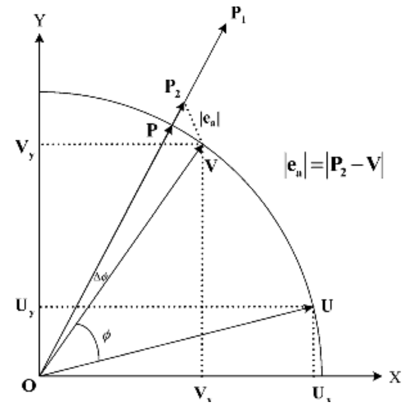

Fig. 13. Proposed CORDIC operation and approximation error.

A. Approximation Error

Fig. 13 illustrates the CORDIC iteration which consists of a pseudo-micro-rotations and a scaling. In the figure, $\mathbf{U}$ is an input vector to be rotated through angle ϕ. It is assumed that scaling and micro-rotations are implemented in two separate stages. $\mathbf{P_1}$ be the rotated vector after m_1 micro-rotations given by

$$\mathbf{P_1} = \prod_{i=0}^{m_1-1} \mathbf{R}(i)\mathbf{U}. \tag{11}$$

The rotation matrix $\mathbf{R}(i)$ is given by (3). The ith scaling factor is given by

$$S(i) \triangleq 1 + \delta_i 2^{-s(i)} \tag{12}$$

such that after m_2 iterations of scaling we get

$$\mathbf{P_2} = \prod_{i=0}^{m_2-1} S(i)\mathbf{P_1}. \tag{13}$$

After the micro-rotations, there is a discrepancy $\Delta\phi$ between the desired angle and the resultant angle due to the limited number of micro-rotations. Moreover, $\mathbf{P_1}$ cannot reach $\mathbf{P}$ on the circle after the scaling since K_A is an approximated value which is not same as the required K given as (4). Similar to the method used in [20], the approximation error is evaluated as a distance between the desired output $\mathbf{V}$ and the actual CORDIC output $\mathbf{P_2}$ as follows:

$$\mathbf{e}_a = |\mathbf{V} - \mathbf{P_2}|. \tag{14}$$

Without loss of generality, $\Delta\phi$ is assumed to be greater than zero and K_A/K is greater than one as shown in Fig. 13. Then

$$|\mathbf{e}_a|^2 = |\mathbf{V}|^2 + K_A^2|\mathbf{P_1}|^2 - 2K_A|\mathbf{V}||\mathbf{P_1}|\cos\Delta\phi. \tag{15}$$

If $\Delta\phi$ is sufficiently small

$$|\mathbf{e}_a|^2 \simeq (|\mathbf{V}| - K_A|\mathbf{P_1}|)^2 + K_A|\mathbf{V}||\mathbf{P_1}|\Delta\phi^2. \tag{16}$$

Since $|\mathbf{V}| = |\mathbf{U}| = K|\mathbf{P_1}|$

$$|\mathbf{e}_a|^2 \simeq \left(\left(1 - \frac{K_A}{K}\right)^2 + \Delta\phi^2\right)|\mathbf{U}|^2. \tag{17}$$

For the known and fixed angle, an expectation of the approximation error can be estimated once we know the input statistics as

$$E(|\mathbf{e}_a|^2) = \left(\left(1 - \frac{K_A}{K}\right)^2 + \Delta\phi^2\right) E(|\mathbf{U}|^2). \tag{18}$$

It can be seen that the accuracy of CORDIC depends on how closely the angle difference $\Delta\phi$ approximates to zero, and also the ratio of scale-factors K_A/K approximates to one.

B. Round-off Error

As the CORDIC iteration progresses through shift-add operations, the word-length increases, and consequently requires rounding after each CORDIC iteration. Let $\mathbf{e}_r$ be the round-off error. The magnitude of round-off error depends on the word-length in a data-path, especially the length of fractional bits which is denoted as b. The mean and variance of $\mathbf{e}_r$ are estimated separately and added to obtain $E(|\mathbf{e}_r|^2)$ as

$$E(|\mathbf{e}_r|^2) = \left|E(\mathbf{e}_r)\right|^2 + \mathrm{Var}(\mathbf{e}_r(0)) + \mathrm{Var}(\mathbf{e}_r(1)) \tag{19}$$

where $\mathbf{e}_r = [\mathbf{e}_r(0)\, \mathbf{e}_r(1)]^T$. When a data with b fractional bits is shifted by i, the mean and variance of resultant round-off error are calculated in [21] as

$$M_{b,i} = 2^{-b-1}(2^{-i} - 1) \tag{20a}$$

$$\mathrm{V}_{b,i} = \frac{2^{-2b}}{12}(1 - 2^{-2i}) \tag{20b}$$

respectively. The round-off error generated from each micro-rotation and scaling is propagated forward through the CORDIC iterations, and get accumulated in the output vector $\mathbf{P_2}$. The magnitude of accumulated round-off error in the estimation of vector $\mathbf{P_1}$ after m_1 micro-rotations is

$$E(\mathbf{e}'_r) = \sum_{i=0}^{m_1-1} \prod_{j=i+1}^{m_1-1} \mathbf{R}(j) \begin{bmatrix} -\sigma_i \mathrm{M}_{b,k(i)} \\ \sigma_i \mathrm{M}_{b,k(i)} \end{bmatrix} \tag{21a}$$

$$\begin{aligned} \mathrm{Var}\big(\mathbf{e}'_r(0)\big) &= \mathrm{Var}\big(\mathbf{e}'_r(1)\big) \\ &= \sum_{i=0}^{m_1-1} \prod_{j=i+1}^{m_1-1} \det\mathbf{R}(j) \mathrm{V}_{b,k(i)}. \end{aligned} \tag{21b}$$

The final round-off error accumulated in the output vector $\mathbf{P_2}$ after scaling is calculated by using (22)

$$\begin{aligned} |E(\mathbf{e}_r)|^2 = &\left(K_A E\big(\mathbf{e}'_r(0)\big) + \sum_{i=0}^{m_2-1} \prod_{j=i+1}^{m_2-1} S(j)\mathrm{M}_{b,s(i)}\right)^2 \\ &+ \left(K_A E\big(\mathbf{e}'_r(1)\big) + \sum_{i=0}^{m_2-1} \prod_{j=i+1}^{m_2-1} S(j)\mathrm{M}_{b,s(i)}\right)^2 \end{aligned} \tag{22a}$$

$$\begin{aligned} \mathrm{Var}(\mathbf{e}_r(0)) = \mathrm{Var}(\mathbf{e}_r(1)) = &K_A \mathrm{Var}\big(\mathbf{e}'_r(0)\big) \\ &+ \sum_{i=0}^{m_2-1} \prod_{j=i+1}^{m_2-1} |S(j)| \mathrm{V}_{b,s(i)}. \end{aligned} \tag{22b}$$

TABLE IV
MINIMUM CLOCK PERIOD OF DIFFERENT ARCHITECTURES

designs	clock period	
	accuracy level-1	accuracy level-2
conventional CORDIC	$T_A + T_{FF} + 5T_{MX}$	$T_A + T_{FF} + 6T_{MX}$
reference design	$T_A + T_{FF} + 5T_{MX}$	$T_A + T_{FF} + 6T_{MX}$
interleaved scaling	$T_A + T_{FF} + 5T_{MX}$	$T_A + T_{FF} + 5T_{MX}$
separated-scaling	$T_A + T_{FF} + 4T_{MX}$	$T_A + T_{FF} + 5T_{MX}$
single-rotation cascade	T_A	T_A
bi-rotation cascade	$T_A + T_{FF} + 2T_{MX}$	$T_A + T_{FF} + 2T_{MX}$

C. Error of the Proposed CORDIC

For the case of $m_1 = 4$ and $m_2 = 3$, all the necessary values in (18) and (22) can be obtained from Tables I and III. Additionally, the number of fractional bit b and average power $E(|\mathbf{U}|^2)$ are needed for the estimation of round-off and approximation error, respectively. Equation (22) is valid only for the case when the micro-rotations and scaling are performed in two separated stages. If the micro-rotations and scaling are deployed in interleaved manner, the sequence of $\mathbf{R}(i)$ and $S(i)$ in (21) and (22) should be changed accordingly in order to represent the transfer function in interleaved manner.

If we want to reduce the total error, and the approximation error is dominant error source, it would be a better strategy to increase the number of micro-rotations and scaling iterations. It would make $\Delta\phi$ and/or $|1 - K_A/K|$ approach to zero. If the round-off error is greater than approximation error, we need to increase the number of fractional bits b in order to reduce the total error.

VI. COMPLEXITY CONSIDERATIONS

We discuss here the hardware and time complexities of the proposed design. In the existing literature we do not find similar work on CORDIC implementation of known and fixed rotations. Therefore, we compare the proposed design with the conventional CORDIC design for the rotation of unknown angle. We have used the basic CORDIC processor in [3, Fig. 2] for the implementation of conventional CORDIC. In addition, we have designed a reference architecture (see Fig. 1) for straightforward implementation of fixed rotations, and we have compared the complexities and speed performance of the proposed design with the conventional and reference design. The maximum deviation of ϕ amounting to $\sim\ 0.04°$ is assumed to be accuracy level-1 (AL-1) and that amounting to $\sim\ 0.0005°$ is assumed to be accuracy level-2 (AL-2), so that AL-1 and AL-2 would correspond to the proposed CORDIC implementations of rotation through four and six micro-rotations, respectively.

A. Complexity of the Conventional and Reference CORDIC

The conventional rotation-mode CORDIC requires three L-bit adders, three L-bit registers, two barrel-shifters and four MUXes, where L is the word-length. The complexity of barrel-shifter, however, depends on the accuracy of implementation. Considering that the rotations are mapped to the first quadrant, the conventional CORDIC would involve 11 iterations and 17 iterations, respectively, for AL-1 and AL-2. Each of its pair of barrel-shifters would thus involve four and five stages, where each stage requires L 2:1 MUXes, for AL-1

TABLE V
AREA AND TIME COMPLEXITIES OF DIFFERENT ARCHITECTURES

complexity	conventional CORDIC		reference design[1]		interleaved-scaling[2]		separated-scaling[3]		single-rotation cascade		bi-rotation cascade	
	AL-1	AL-2	AL-1	AL-2	AL-1	AL-2	AL-1	AL-2	AL-1	AL-2	AL-1	AL-2
L-bit adder	3	3	2	2	2	2	4	4	14	22	7	11
register (bits)	$3L$	$3L$	$2L+10$	$2L+16$	$2L+8$	$2L+12$	$4L+7$	$4L+11$	$12L$	$20L$	$8L+9$	$12L+15$
2:1 MUX	$12L$	$14L$	$10L$	$12L$	$10L$	$10L$	$14L$	$20L$	0	0	$13L$	$21L$
ROM (bits)	$11L$	$17L$	0	0	28	44	18	44	0	0	0	0
tri-state buffer	0	0	0	0	$4L$	$4L$	0	0	0	0	0	0
throughput	1/11	1/17	1/10	1/16	1/7	1/11	1/4	1/6	1	1	1/2	1/2
latency	11	17	10	16	7	11	7	11	7	11	7	11

L is word-length. AL-1 and AL-2 stand for accuracy-level-1 and 2, respectively, which correspond to error of 0.04 deg and 0.5×10^{-3} deg. (1) Reference design and (2) Interleaved scaling are implemented by the circuits of Figs.1. and 8, respectively. (3) Separated scaling is implemented by the circuit of Fig.2 for micro-rotations and that of Fig.10 for scaling.

TABLE VI
AREA-TIME COMPLEXITIES OF DIFFERENT ARCHITECTURES BASED ON SYNTHESIS RESULT USING TSMC 90-nm LIBRARY

designs	accuracy level-1						accuracy level-2					
	area	clock	TPT	latency	ACT	ADP	area	clock	TPT	latency	ACT	ADP
conventional CORDIC	2987	2.52	36.07	27.72	27.72	82799	6545	4.24	13.87	72.08	72.08	471763
reference design	2471	2.49	40.16	24.90	24.90	61527	5323	4.22	14.81	67.52	67.52	359408
interleaved-scaling	2674	2.56	55.8	17.92	17.92	47918	5320	4.18	21.74	45.98	45.98	244613
separated-scaling	4074	2.29	109.17	16.03	9.16	37317	9508	4.17	39.96	45.87	25.02	237890
single-rotation cascade	6920	1.81	552.48	12.67	1.81	12525	23070	3.58	279.32	39.38	3.58	82590
bi-rotation cascade	5819	2.21	226.24	15.47	4.42	25719	17997	3.96	126.26	43.56	7.92	142536

TPT stands for throughput and calculated per micro-second. ACT stands for average computation time measured in nano-seconds. ADP stands for area-delay product, calculated as the product of area in sq.um and ACT in nanoseconds. Word-size $L = 16$ and 32 are, respectively, used for accuracy level-1 and -2.

TABLE VII
RELATIVE ADVANTAGES OF PROPOSED DESIGNS OVER THE REFERENCE DESIGN FOR FIXED ROTATIONS

parameters	single-rotation cascade		bi-rotation cascade	
	AL-1	AL-2	AL-1	AL-2
area	2.80	4.33	2.35	3.38
clock period	−27.30	−15.17	−11.24	−6.16
throughput	13.76	18.86	5.63	8.53
latency	−1.97	−1.71	−1.61	−1.55
area-delay	−4.91	−4.35	−2.39	−2.52

Except the clock period all other entries in this table are in number of times. Positive sign implies that the value of the parameter for the proposed design is higher than that of the reference design and negative sign implies it lower.

and AL-2, respectively. Apart from that, all the three input registers are to be loaded through MUXes to allow direct input as well as the input through the feedback path. The ROM needs to store L bits $\arctan$ angles for 11 and 17 iterations for AL-1 and AL-2, respectively. The duration of minimum clock period in conventional CORDIC is $T = T_A + T_{FF} + 5T_{MX}$ and $T_A + T_{FF} + 6T_{MX}$ for AL-1 and AL-2, where T_A, T_{FF}, and T_{MX} are the L-bit addition-time, D flip flop delay and delays of 2:1 MUX, respectively.

The reference CORDIC for the fixed rotation (shown in Fig. 1), consists of two adders, two barrel-shifters, one sign-bit-register and two input registers with MUXes. The MUXes accompanied by the input registers are, however, not shown in the reference as well as the proposed designs (as discussed in Section II for the description of Fig. 1). We assume that the rotation is mapped to half quadrant range so that for accuracy of AL-1 and AL-2, it requires 10 and 16 iterations. It has the same barrel-shifter complexity and time-complexity as the conventional CORDIC.

B. *Complexity of the Proposed CORDIC Designs*

Each of the proposed CORDIC designs involves a latency of 7 cycles and 11 cycles for accuracy level-1 and 2, respectively. But the hardware requirement, duration of clock period and throughput rate differ from one another. We discuss these complexities of proposed CORDIC designs in four categories: 1) single CORDIC cell with interleaved-scaling; 2) single CORDIC cell with separate-scaling; 3) single-rotation cascade; and 4) bi-rotation cascade.

1) Cordic Cell With Interleaved Scaling and Micro-Rotations: As shown in Fig. 8, the CORDIC implementation by interleaved scaling requires an additional ROM and a line changer over that of reference design of Fig. 1. The line changer requires $4L$ number of tri-state buffers and a T flip flop to generate the control-bit. Using hardwired pre-shifting, each of the pair of barrel-shifters involves 4 stages of 2:1 MUXes for implementing all the necessary shifts for micro-rotations as well as scaling for both accuracy levels. Accordingly, the duration of minimum clock period for the proposed interleaved CORDIC can be found to be $T = T_A + T_{FF} + 5T_{MX}$ for both the accuracy levels. It involves 7 and 11 iterations for AL-1 and AL-2, respectively, to implement both scaling and micro-rotations. The ROM therefore needs to store 7 and 11 control words of 4-bit size to be used by the barrel-shifter, and the SBR is of 7 and 11 bit size for AL-1 and AL-2, respectively.

2) CORDIC Cell With Separate Scaling and Micro-Rotation Stages: CORDIC implementation of fixed rotation could be performed in two pipelined stages, where micro-rotations are implemented by Fig. 2 and scaling is implemented by Fig. 7.

TABLE VIII
AREA AND TIME COMPLEXITIES COMPARISON OF DIFFERENT ARCHITECTURES ON FPGA

designs	Xilinx Spartan-3A DSP (XC3SD1800A-4FG676)						Xilinx Virtex-4 (XC4VSX35-10FF668)					
	accuracy level-1			accuracy level-2			accuracy level-1			accuracy level-2		
	NOS	MUF	SDP	NOS	MUF	SDP	NOS	MUF	SDP	NOS	MUF	SDP
conventional CORDIC	178	113.3	17.28	382	93.5	69.45	169	180.9	10.27	383	149.9	43.43
reference design	144	113.3	12.70	297	93.5	50.82	135	180.9	7.46	297	149.9	31.70
interleaved-scaling	122	105.9	8.06	257	93.7	30.17	125	169.5	5.16	252	148.2	18.70
separated-scaling	219	104.9	8.35	560	80.9	41.53	198	185.7	4.26	555	131.0	25.41
single-rotation cascade	130	228.7	0.56	398	184.8	2.15	130	398.3	0.32	389	310.3	1.25
bi-rotation cascade	279	116.3	4.79	911	97.5	18.68	279	196.8	2.83	906	153.1	11.83

NOS stands for the number of slices. MUF stands for maximum operating frequency in MHz. SDP stands for slice-delay product.

Using hardwired pre-shifting, the barrel-shifter involves 3 and 4 stages of 2:1 MUXes to implement the necessary shifts for micro-rotation and 2 and 4 stages for scaling for accuracy levels-1 and -2, respectively. The ROM therefore needs to store 4 control words of 3 bit size for micro-rotation and 3 control words of 2 bit size for scaling to be used by the barrel-shifter for AL-1 and 11 control words of 4 bit size for AL-2, along with SBR of 7 and 11 bit size for AL-1 and AL-2, respectively. Accordingly, the duration of minimum clock period for this implementation is found to be $T = T_A + T_{FF} + 4T_{MX}$ and $T = T_A + T_{FF} + 5T_{MX}$ for both accuracy levels-1 and 2, respectively. Although it involves 3 and 5 iterations for scaling, it involves 4 and 6 iterations for micro-rotations for AL-1 and AL-2, respectively. Therefore, the iteration count in this case is 4 and 6 for these two cases.

3) Single-Rotation Cascade: The single-rotation cascaded CORDIC for fixed-angle rotation is shown in Fig. 5. For accuracy level-1 it involves seven stages out of which four stages perform the micro-rotations and the three remaining stages perform scaling. The rotation modules are modified to implement shift operations for scaling. Each stage requires two adders and two pipelining registers (except that the last stage does not require pipeline register). All the shiftings are hardwired and there is no feed-back path in this circuit. Therefore, it does not require any ROM, SBR, barrel-shifters or MUXes. The duration of minimum clock period for this implementation is $T = T_A$ for both accuracy levels and produces one output in each cycle.

4) Bi-Rotation Cascade: For accuracy level-1, it requires a cascaded two-stage bi-rotation CORDIC as shown in Fig. 6 for micro-rotation. To implement scaling, it requires a two-factor scaling circuit of Fig. 9 and time-multiplexed circuit of Fig. 11 for one-factor scaling. For accuracy level-2, it requires a cascaded three-stage bi-rotation CORDIC (see Fig. 6) for micro-rotation. To implement scaling, it requires three cascaded stages consisting of two stages of two-factor scaling circuit of Fig. 9 and one stage of a time-multiplexed circuit of Fig. 11. The duration of minimum clock period for both the accuracy-levels is $T = T_A + T_{FF} + 2T_{MX}$ and it gives an output in every alternate cycle.

C. Comparative Performances

The expressions of clock periods of the architectures are listed in Table IV. The single-rotation CORDIC has the minimum of clock period of one addition-time and bi-rotation CORDIC has slightly higher clock period. The hardware and time-complexities of different architectures are listed comprehensively in Table V. The CORDIC algorithms are written in hardware description language and synthesized by Synopsys Design Compiler using the TSMC 90-nm library to obtain the complexities of proposed and the reference designs. Word size $L = 16$ and 32 are used for accuracy level-1 and -2, respectively. The area, clock period, latency, throughput, average computation time (ACT), and area-delay product (ADP) are listed in Table VI.

The reference design has the same clock-period as the conventional CORDIC but yields $\sim 9\%$ more throughput and involves $\sim 18\%$ less area, $\sim 8\%$ less latency and $\sim 25\%$ less area-delay product (ADP), over the conventional one, in average over the two levels of accuracy. The proposed design of single CORDIC cell with interleaved-scaling has $\sim 4\%$ more area but offers $\sim 43\%$ more throughput and involves $\sim 30\%$ less latency and $\sim 27\%$ less ADP, in average over both the levels of accuracy, compared to the reference design. The proposed design of single CORDIC unit with separate-scaling similarly, has nearly $\sim 72\%$ more area but offers nearly 2.7 time the throughput and involves $\sim 37\%$ less ADP and two-third of the latency over the reference design. The relative advantages of single-rotation cascade and bi-rotation cascade are shown in Table VII. In average over both the levels of accuracy, the single-rotation and bi-rotation cascades, respectively, involve nearly 3.6 times and 2.9 times more area over the reference design, but offer nearly 16.3 times and 7.0 times more throughput, and involve 4.6 and 2.5 times less ADP with nearly half and two-third less latency over the other.

The reference and proposed designs are also implemented on the field-programmable gate-array (FPGA) platform of Xilinx devices. The number of slices (NOS), maximum operating frequency (MUF), and slice-delay product (SDP) using two different devices of Spartan-3A (XC3SD1800A-4FG676) and Virtex-4 (XC4VSX35–10FF668) are listed in Table VIII. The proposed design of single-rotation cascade involves smaller number of slices and faster operating frequency over the conventional and reference designs for two devices. In average over both levels of accuracy and devices, the single-rotation cascade offers nearly 23.5× and 32.2× less SDP over the reference design and conventional design, respectively.

VII. CONCLUSION

The number of micro-rotations for rotation of vectors through known and fixed angles are optimized and several possible dedicated circuits are explored for rotation-mode CORDIC processing with different levels of accuracy. The proposed CORDIC cell with interleaved scaling involves $\sim 4\%$ more area, but offers $\sim 43\%$ more throughput and involves nearly 30% less latency and $\sim 20\%$ less ADP, than the reference design for known and fixed rotations. The proposed single-rotation cascade and birotation cascade require, respectively, ~ 3.6 and ~ 2.9 times more area over the reference design, but offer nearly 16.3 and 7.0 times more throughput, and involve nearly 4.6 and 2.5 times less ADP with nearly half and two-thirds of the latency of the other. With progressing scaling trends, since the silicon area is getting continually cheaper, it appears to be a good idea to use the cascaded designs for their potential for high-throughput and low-latency implementation. It is found that higher accuracy could be achieved in case of smaller angles of rotation when the same number of micro-rotations are used. The small angle rotators could therefore be very much useful for shape design and curve tracing for animation and gaming devices. The fixed-angle CORDIC rotation would have wide applications in signal processing, games, animation, graphics and robotics, as well.

References

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

[2] J. S. Walther, "A unified algorithm for elementary functions," in *Proc. 38th Spring Joint Comput. Conf.*, 1971, pp. 379–385.

[3] Y. H. Hu, "CORDIC-based VLSI architectures for digital signal processing," *IEEE Signal Process. Mag.*, vol. 9, no. 3, pp. 16–35, Jul. 1992.

[4] P. K. Meher, J. Valls, T.-B. Juang, K. Sridharan, and K. Maharatna, "50 years of CORDIC: Algorithms, architectures and applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 9, pp. 1893–1907, Sep. 2009.

[5] Y. H. Hu and S. Naganathan, "An angle recoding method for CORDIC algorithm implementation," *IEEE Trans. Comput.*, vol. 42, no. 1, pp. 99–102, Jan. 1993.

[6] Y. H. Hu and H. H. M. Chern, "A novel implementation of CORDIC algorithm using backward angle recoding (BAR)," *IEEE Trans. Comput.*, vol. 45, no. 12, pp. 1370–1378, Dec. 1996.

[7] C.-S. Wu, A.-Y. Wu, and C.-H. Lin, "A high-performance/low-latency vector rotational CORDIC architecture based on extended elementary angle set and trellis-based searching schemes," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 50, no. 9, pp. 589–601, Sep. 2003.

[8] T.-B. Juang, S.-F. Hsiao, and M.-Y. Tsai, "Para-CORDIC: Parallel cordic rotation algorithm," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 8, pp. 1515–1524, Aug. 2004.

[9] T. K. Rodrigues and E. E. Swartzlander, "Adaptive CORDIC: Using parallel angle recoding to accelerate CORDIC rotations," in *Proc. 40th Asilomar Conf. Signals, Syst. Comput. (ACSSC)*, 2006, pp. 323–327.

[10] K. Maharatna, S. Banerjee, E. Grass, M. Krstic, and A. Troya, "Modified virtually scaling free adaptive CORDIC rotator algorithm and architecture," *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 15, no. 11, pp. 1463–1474, Nov. 2005.

[11] C. Y. Kang and E. E. Swartzlander, Jr., "Digit-pipelined direct digital frequency synthesis based on differential CORDIC," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 5, pp. 1035–1044, May 2006.

[12] L. Vachhani, K. Sridharan, and P. K. Meher, "Efficient CORDIC algorithms and architectures for low area and high throughput implementation," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 56, no. 1, pp. 61–65, Jan. 2009.

[13] B. G. Blundell, *An Introduction to Computer Graphics and Creative 3-D Environments*. London, U.K.: Springer-Verlag, 2008.

[14] T. Lang and E. Antelo, "High-throughput CORDIC-based geometry operations for 3D computer graphics," *IEEE Trans. Comput.*, vol. 54, no. 3, pp. 347–361, Mar. 2005.

[15] M.-P. Cani, T. Igarashi, and G. Wyvill, *Interactive Shape Design*. San Rafael, CA: Morgan & Claypool, 2007.

[16] K. J. Jones, "2D systolic solution to discrete Fourier transform," *IEE Proc. ∼ Comput. Digit. Techn.*, vol. 136, no. 3, pp. 211–216, May 1989.

[17] P. K. Meher, J. K. Satapathy, and G. Panda, "Efficient systolic solution for a new prime factor discrete Hartley transform algorithm," *IEE Proc. Circuits, Devices Syst.*, vol. 140, no. 2, pp. 135–139, Apr. 1993.

[18] S. Freeman and M. O'Donnell, "A complex arithmetic digital signal processor using cordic rotators," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 1995, pp. 3191–3194.

[19] J. R. Cavallaro and F. T. Luk, "CORDIC arithmetic for a SVD processor," *J. for Parallel Distrib. Comput.*, vol. 5, pp. 271–290, 1988.

[20] C. H. Lin and A. Y. Wu, "Mixed-scaling-rotation CORDIC (MSR-CORDIC) algorithm and architecture for high-performance vector rotational DSP applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 11, pp. 2385–2396, Nov. 2005.

[21] S. Y. Park and N. I. Cho, "Fixed-point error analysis of CORDIC processor based on the variance propagation formula," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 3, pp. 573–584, Mar. 2004.

[22] A. R. Omondi, *Computer Arithmetic Systems: Algorithms, Architectures, and Implementations*. New York: Prentice-Hall, 1994.

[23] A. K. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[24] *Signal Compression : Coding of Speech, Audio, Text, Image And Video, ser. Selected Topics in Electronics and Systems*, N. E. Jayant, Ed. Singapore: World Scientific, 1997, vol. 9.

Pramod Kumar Meher (SM'03) is currently working as a Senior Scientist with the Institute for Infocomm Research, Singapore. His research interests include design of dedicated and reconfigurable architectures for computation-intensive algorithms pertaining to signal, image, and video processing, communication, bio-informatics, and intelligent computing. He has contributed more than 180 technical papers to various reputed journals and conference proceedings.

Dr. Meher is a Fellow of the Institution of Electronics and Telecommunication Engineers, India. He has served as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: EXPRESS BRIEFS during 2008–2011. Currently, he is serving as a speaker for the Distinguished Lecturer Program (DLP) of IEEE Circuits Systems Society, and Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART I: REGULAR PAPERS, IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, and *Journal of Circuits, Systems, and Signal Processing*. He was a recipient of the Samanta Chandrasekhar Award for excellence in research in engineering and technology for the year 1999.

Sang Yoon Park (S'03–M'11) received the B.S., M.S., and Ph.D. degrees from Seoul National University, Seoul, Korea, in 2000, 2002, and 2006, respectively.

He was a Research Fellow with Nanyang Technological University, Singapore, from 2007 to 2008. He joined the Institute for Infocomm Research, Singapore, in 2008, where he is currently a Research Scientist. His research interests include architectures and algorithms for low-power/high-performance digital signal processing and communication systems.