δ-error elimination using truncation retention method and implementation on FPGA for non-redundant CORDIC

Yang Gao1*, Jian Liu1, Zhiwu Zhang1
1No.724 Research Institute of CSIC, ChangQing Road 30, JiangNing District, Nanjing City, Jiangsu Province, 211106, People’s Republic of China  
*gaoyang_3006@163.com

Abstract. In this paper, δ-error is concerned and eliminated in digital implementation of CORDIC algorithm. An analysis on the accumulation mechanism of the computing error which leads to δ-errors is first conducted. And a truncated bits retention method is proposed to compensate the accumulated computing errors and eliminate δ-errors. Results of numerical simulations show that δ-error elimination and calculation accuracy improvement are achieved with the increase of SY even for extremely small inputs. Hardware implementation and resource consumption are also discussed.

1. Introduction
Coordinate rotation digital computer (CORDIC) is a high-performance algorithm to realize a variety of elementary functions [1, 2] and has been widely researched and applied in digital signal processing area [3-5]. In a digital implementation of CORDIC algorithm, δ-errors are easily occurred because of the unavoidable quantization errors, and lead to unacceptable results especially when the input amplitudes are small [6-8]. Although computation error can be decreased by increasing the number of iterations and the number of digital representation bits, it is very resource-intensive. Resource-saving δ-error elimination method has not been revealed.

In 1990s, redundant arithmetic was introduced to speed up the algorithm in hardware implementation [9]. The iteration formula of non-redundant (NR) CORDIC, which is advanced from the traditional redundant (TR) CORDIC, can be rewritten with a variable magnification factor SY as:

\[
\begin{align*}
X_{i+1} &= X_i - \delta_i \cdot \Delta Y_i \cdot 2^{-(2i-1+SY)} \\
\Delta Y_{i+1} &= 2 \cdot \Delta Y_i + \delta_i \cdot X_i \cdot 2^{2+SY} \\
Z_{i+1} &= Z_i - \delta_i \cdot a_i
\end{align*}
\]

where \(i\) is the index of iteration, \(i = 1, 2, 3, \ldots, N\), and \(N\) is the number of iterations; \(a_i = \arctan(2^{i-1})\). Variable \(Z_i\) accumulates the micro-rotation angles according to rotation direction \(\delta_i\). In TR CORDIC, \(\delta_i\) is selected from a redundant set \{-1, 0, 1\} by a complicated select function. However, in this NR CORDIC, \(\delta_i\) is selected from the set \{-1, 1\} by the sign of \(\Delta Y_i\) as applied in basic (BS) vectoring mode CORDIC [1, 2], i.e., \(\delta_i = -1\) if \(\Delta Y_i \geq 0\), \(\delta_i = 1\) if others. In NR CORDIC, \(Y_i\) is magnified according to the iteration index \(i\) and \(SY\) as \(\Delta Y_i = 2^{i+SY} Y_i\). In TR CORDIC, the value \(SY\) is fixed to \(-1\).

The rest of the paper is organized as follows. First, an analysis on the accumulation mechanism of computing error which leads to δ-errors in NR CORDIC is conducted, and a truncation retention
method is proposed to compensate the accumulation error with a simplified compensation formula. Then, numerical simulations and results are presented followed by the hardware implementation and resource consumption of truncation retention NR CORDIC with different SY. At last, a summary is given.

2. Error accumulation and compensation in NR CORDIC

Different from previous error analysis which is aimed to help researchers to find the error bounds of CORDIC [7], our analysis on the accumulation mechanism of computing error is to find the source and the elimination method of \( \delta \)-error. Here, only the computing error introduced during iterations is concerned.

Assume \( X_i = x_i + e_{x_i} \) and \( \Delta Y_i = \Delta y_i + e_{\Delta y_i} \), where \( x_i \) and \( \Delta y_i \) are the actual values which respectively represent the true values \( X_i \) and \( \Delta Y_i \) in a finite representation precision system, while \( e_{x_i} \) and \( e_{\Delta y_i} \) are the corresponding rounding errors. Equation (1) is rewritten as

\[
\begin{align*}
X_{i+1} &= x_i + e_{x_i} = (x_i + e_{x_i}) - \delta_i \cdot (\Delta y_i + e_{\Delta y_i}) \cdot 2^{-(i+1+SY)} \\
\Delta Y_{i+1} &= 2 \cdot (\Delta y_i + e_{\Delta y_i}) + \delta_i \cdot (x_i + e_{x_i}) \cdot 2^{2+SY}.
\end{align*}
\]

Set

\[
\Delta Y_i \cdot 2^{-(i+1+SY)} = \text{Round} \left\{ \Delta Y_i \cdot 2^{-2(i+SY)} \right\} + e_{\Delta y_i} \cdot 2_i,
\]

where \( \text{Round} \{ \cdot \} \) denotes the rounding to the nearest integer operation, and \( e_{\Delta y_i} \cdot 2_i \) represents the rounding error of \( \Delta y_i \cdot 2^{-(i+1+SY)} \) in a finite precision system. Therefore, the iteration formula of NR CORDIC can be separated into two parts. One is the part that is realizable in a finite precision system, written as

\[
\begin{align*}
x_{i+1} &= x_i - \delta_i \cdot \text{Round} \left\{ \Delta Y_i \cdot 2^{-2(i+SY)} \right\} \\
\Delta Y_{i+1} &= 2 \cdot \Delta y_i + \delta_i \cdot x_i \cdot 2^{2+SY}.
\end{align*}
\]

The other is the rounding error accumulation part, written as

\[
\begin{align*}
ex_{i+1} &= e_{x_i} - \delta_i \cdot \left( e_{\Delta y_i} \cdot 2_i + e_{\Delta y_i} \cdot 2^{-(i+1+SY)} \right) \\
e_{\Delta y_{i+1}} &= 2 \cdot e_{\Delta y_i} + \delta_i \cdot e_{x_i} \cdot 2^{2+SY}
\end{align*}
\]

In an actual realization of equation (4) without initial input error, the only computing error source is the rounding of \( \Delta y_i \cdot 2^{-(i+1+SY)} \), i.e., \( e_{\Delta y_i} \cdot 2_i \), which is magnified and accumulated implicitly and unavoidably during iterations as shown in equation (5) and consequently leads to \( \delta \)-errors.

Since error is magnified and accumulated iteration by iteration, compensation can be implemented iteration by iteration. However, unless it is artificially magnified with a proper magnification, the rounding error \( e_{x_i} \) is too small to be represented in the finite precision system. Since variable \( x_i \) used in the \( \Delta y \) recurrence is magnified by \( 2^{2+SY} \) in equation (4), \( 2^{2+SY} \) is chosen as the proper magnification for \( e_{x_i} \). Hence, equation (5) is rewritten as

\[
\begin{align*}
\Delta ex_{i+1} &= \Delta e_{x_i} - \delta_i \cdot \left( e_{\Delta y_i} \cdot 2_i \cdot 2^{2+SY} + e_{\Delta y_i} \cdot 2^{-(i+1+2)} \right) \\
e_{\Delta y_{i+1}} &= 2 \cdot e_{\Delta y_i} + \delta_i \cdot \Delta e_{x_i}
\end{align*}
\]

where \( \Delta e_{x_i} = 2^{2+SY} \cdot e_{x_i} \).

Furthermore, after error compensation is implemented, error \( e_{\Delta y_i} \) is going to be very small and can be removed from equation (6). Therefore, a simplified error formula can be obtained as

\[
\Delta ex_{i+1} = \Delta e_{x_i} - \delta_i \cdot e_{\Delta y_i} \cdot 2^{2+SY}.
\]

Finally, this simplified \( \Delta e_{x_i} \) is compensated back into the realizable part in equation (4). In a summary, this simplified error compensation, named as truncation retention method, can be expressed as
\begin{align}
    x_{i+1} &= x_i - \delta_i \cdot \text{Round} \left\{ \Delta y_i \cdot 2^{-(2i-1+SY)} \right\} \\
    \Delta y_{i+1} &= 2 \cdot \Delta y_i + \delta_i \cdot \left( x_i \cdot 2^{2+SY} + \Delta e_x \right) \\
    z_{i+1} &= z_i - \delta_i \cdot a_i \\
    \Delta e_{x,i+1} &= \Delta e_x - \delta_i \cdot \Delta e y 2_i
\end{align}

where \( \Delta e y \text{Round} \left\{ \Delta y_i \cdot 2^{-(2i-1+SY)} \right\} \), which retains the first \((2+SY)\) Most Significant Bits (MSB) from the truncated bits during the rounding of \( \Delta y_i \cdot 2^{-(2i-1+SY)} \). In this truncation retention method, value \( \Delta e_x \) is utilized for compensation and can be achieved easily in a finite precision system on digital signal processing (DSP) chips, such as field programmable gate array (FPGA). It is obvious that the compensation value is increased with the increase of \( SY \). Consequently, the \( \delta \)-error elimination ability is improved, which is verified in the following numerical simulations section.

3. Numerical simulations

Numerical simulations are carried out on fixed-point BS CORDIC, uncompensated NR CORDIC and truncation retention NR CORDIC with different \( SY \). The number of bits for digital representation \( b \) is 16, the number of iterations \( N \) is 14. The 100 set of input vector \((X_i, Y_i)\) are formed by two integers selected from 1 to 10. The amplitudes of these input vectors are extremely small compared to the maximum positive integer \( (2^b-1) \). An extra double-precision floating-point BS CORDIC is simulated in MATLAB for reference. Figure 1 shows the number of \( \delta \)-errors at different iteration stages in different CORDIC implementation methods. As we can see, with the increase of \( SY \), truncation retention NR CORDIC gradually improves the \( \delta \)-error elimination ability in two remarkable patterns. One is that the appearing of \( \delta \)-error is forced backward to the latter iterations. The other is that the total number of \( \delta \)-errors is reducing. As a fact, when \( SY = 5 \), the \( \delta \)-error is completely eliminated, which is not shown in figure 1.

Angle calculation errors in truncation retention NR CORDIC is displayed in figure 2. To display clearly, only part of the results are depicted. In fact, the maximum angle calculation error is about 6.3° in Fixed-point BS CORDIC, and about 1.75° for \( SY = -2 \) in truncation retention NR CORDIC. As shown in figure 2, the maximum error is reduced to 0.22° for \( SY = 0 \), to 0.06° for \( SY = 2 \), and to 0.02° for \( SY = 4 \).

![Figure 1. The number of \( \delta \)-errors in different CORDIC and different \( SY \).](image-url)
4. Implementations and consumptions in FPGA

The main iterative structure of truncation retention NR CORDIC is depicted in figure 3. The operations of multiplication and division in equation (8) are implemented as left and right shifts, respectively. The ROM (Read-Only Memory) reserves the quantified fixed micro-rotation angles for all iteration stages. The Regs (Registers) are used to streamline the iterations for good timing. Compared to previous structure of TR CORDIC [10], this main structure contains an additional adder, which is marked in figure 3 with thick line, to accomplish the simplified error compensation operation.

The simplified error accumulation scheme is displayed in figure 4. The truncation retention operation is to derive the retained value $\Delta e \Delta y / 2$. The processing procedure of truncation retention operation is described in figure 5. During the program design for a FPGA, the programmer can directly move the wanted bits from one variable to another. This characteristic makes truncation retention operation easily realizable.

---

**Figure 2.** Angle calculation errors in truncation retention NR CORDIC.

**Figure 3.** Main iterative structure of truncation retention NR CORDIC.

**Figure 4.** The simplified error accumulation scheme with truncation retention operation.
Figure 5. Processing procedure of truncation retention operation.

The implementations are carried out on Xilinx’s ISE 12.4 software targeted at a Virtex-5 LX50T FPGA with \( b = 16 \) and \( N = 14 \). Hardware consumptions of uncompensated and truncation retention NR CORDIC are listed in table 1. As we can find, truncation retention NR CORDIC occupies very little extra hardware resource.

|                | NR CORDIC | \( SY \) | -2 | -1 | 0 | 1 | 2 | 3 |
|----------------|-----------|---------|----|----|---|---|---|---|
| Uncompensated  | Registers | 642     | 625| 644| 615| 606| 600|
|                | LUTs      | 788     | 707| 740| 657| 609| 606|
| Truncation retention method | Registers | 642 | 786 | 807 | 773 | 766 | 759 |
|                | LUTs      | 788 | 1014 | 1032 | 934 | 896 | 897 |
| Extra Hardware consumption | Registers | 0 | 161 | 163 | 158 | 160 | 159 |
|                | LUTs      | 0 | 307 | 292 | 277 | 287 | 291 |

5. Conclusions
This paper reveals the \( \delta \)-error elimination ability using a truncation retention method in NR CORDIC. The simulation results show that with the increase of the magnification factor \( SY \), the truncation retention method can effectively reduce and eliminate the \( \delta \)-errors even for extremely small inputs, and achieves a angle calculation error reduction from \( 1.75^\circ (SY = -2) \) to \( 0.02^\circ (SY = 4) \) at a little cost of extra hardware consumption.

References
[1] Volder J. E. (1959) The CORDIC Computing Technique. Electronic Computers IRE Transactions on, EC-8:330–334.
[2] Walther J. S. (1971) A unified algorithm for elementary functions. Spring Joint Computing Conference:272–278.
[3] Pitchandi V., et al. (2012) Bit parallel-iterative circuit for robotic application. IEICE Electronics Express, 9:443–449.
[4] Park S. Y., Yu Y. J. (2012) Fixed-Point Analysis and parameter selections of MSR-CORDIC with applications to FFT designs. IEEE Transactions on Signal Process, 60:6245-6256.
[5] WANG W. T., ZHANG W. X., TAO D. X. (2016) Implementation of FPGA-based CORDIC algorithm used to extract in-pulse characteristics. Modern Electronics Technique, 39:1–11
[6] Hu Y. H. (1992) The quantization effects of the CORDIC algorithm. IEEE Transactions on Signal Processing, 40:834-844.
[7] Hu X. and Bass S. C. (1993) A neglected error source in the CORDIC algorithm. IEEE International Symposium on Circuits and Systems, 1:766–769.
[8] Kota K., Cavallaro J. R. (1993) Numerical accuracy and hardware tradeoffs for CORDIC arithmetic for special-purpose processors. IEEE Transactions on Computers 42:769–779.
[9] Ercegovac M. D., Lang T. (1990) Redundant and on-line CORDIC: application to matrix triangularization and SVD. IEEE Transactions on Computers, 39:725–740.
[10] E. Antelo, et al. (1997) Error analysis and reduction for angle calculation using the CORDIC algorithm. IEEE Transactions on Computers, 46:1264–1271.