

基于旋转模式的改进型 CORDIC 算法研究

张天瑜

(无锡市广播电视大学 机电工程系, 江苏 无锡 214011)

摘 要: 针对 CORDIC 算法的缺陷, 在旋转模式下提出一种改进型 CORDIC 算法, 它不需要查找表和模校正因子, 只需通过简单的移位和加减运算就能实现多种超越函数的计算, 从而能够减少硬件的开销, 提高运算的性能, 并通过重复迭代和区域变换使得该算法能够适用于所有的旋转角度. 误差分析表明该算法具有很小的误差.

关键词: CORDIC 算法; 旋转模式; 麦克劳林级数; 查找表; 模校正因子; 流水线结构

中图分类号: TP301.6

文献标识码: A

文章编号: 1000-7180(2010)03-0093-05

Research of Modified CORDIC Algorithm Based on Rotation Mode

ZHANG Tian-yu

(Department of Mechanical and Electrical Engineering, Wuxi Radio & Television University, Wuxi 214011, China)

Abstract: According to the defects of CORDIC algorithm, a modified CORDIC algorithm is proposed in rotation mode. It needn't look-up table and scale factor correction and can realize the calculation of many transcendental functions only by simple shift and addition-subtraction operation, which can reduce the cost of hardware and improve operational performance. The proposed algorithm is suitable for all the rotation angles by repeated iteration and domain transformation. The error analysis shows that the error of the proposed algorithm is very small.

Key words: CORDIC algorithm; rotation mode; maclaurin series; look-up table; scale factor correction; pipeline structure

1 引言

坐标旋转数字计算机(Coordinate Rotation Digital Computer, CORDIC)算法是在 1959 年由 Jack E. Volder 在设计美国航空导航控制系统的过程中提出的. 在 1974 年提出了能计算出多种超越函数的统一算法. CORDIC 算法的基本思想是通过一系列固定的、逐次递减的、与运算基数相关的基本角度不断偏转从而逼近最终需要达到的旋转角度, 使得向量的旋转和定向运算不需要通过查三角函数表、乘法、开方以及反三角函数等复杂运算. 它的巧妙之处在于基本角度的选取恰好使每次向量以基本角度旋转后, 新向量的坐标计算只需要通过简单的移位和加减运算即可. 由于 CORDIC 算法的复杂度较低、功能完备, 能够运用于向量旋转(复数乘除法)、三角函数、双曲函数、指数函数以及其他一些超越函数的计算, 目前已经被广泛应用于各个方面^[1-3]. 文中从

查找表的移位实现、模校正因子的分解、旋转角度的范围扩展 3 个方面对 CORDIC 算法进行改进, 提出一种改进型 CORDIC 算法, 并设计了流水线结构^[4-5].

2 CORDIC 算法

CORDIC 算法是从一般的向量旋转方程中推导出来的, 它包含三种旋转坐标系, 圆周坐标系、线性坐标系、双曲坐标系, 在每种坐标系下分别有两种工作模式, 旋转模式和向量模式^[6-7]. 文中主要讨论圆周坐标系下的旋转模式. CORDIC 算法的原理图如图 1 所示.

在图 1 中, 根据三角函数理论, 初始向量 $\overrightarrow{OA}$ 逆时针旋转角度到目标向量 $\overrightarrow{OB}$ 可以表示为

$$\begin{cases} x' = x \cos \theta - y \sin \theta = (x - y \tan \theta) \cos \theta \\ y' = y \cos \theta + x \sin \theta = (y + x \tan \theta) \cos \theta \end{cases} \quad (1)$$

若初始向量 $\overrightarrow{OA}$ 顺时针旋转角度 θ 到目标向量

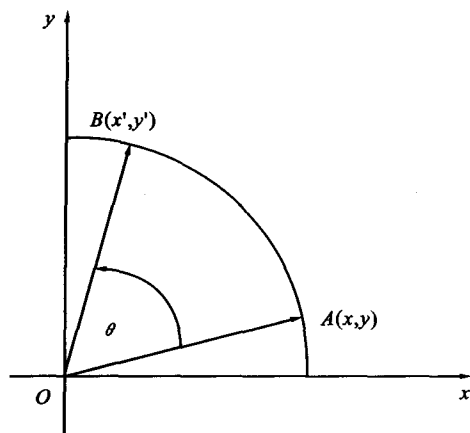


图1 CORDIC 算法的原理图

$\overline{OB}$, 同理可得

$$\begin{cases} x' = x \cos \theta + y \sin \theta = (x + y \tan \theta) \cos \theta \\ y' = y \cos \theta - x \sin \theta = (y - x \tan \theta) \cos \theta \end{cases} \quad (2)$$

CORDIC 算法迭代运算的原理图如图 2 所示。

在图 2 中, 假设 θ 是由连续 n 个微小的基本角度 θ_i 通过旋转叠加而成, 即 $\theta = \sum_{i=0}^{n-1} \sigma_i \theta_i$, $\sigma_i = 1$ 对应逆时针旋转, $\sigma_i = -1$ 对应顺时针旋转。由于计算机在进行计算时采用的是二进制形式, 所以每次旋转的基本角度取 $\theta_i = \arctan(2^{-i})$, 即 $\tan \theta_i = 2^{-i}$ 。根据式(1)和式(2), 向量的旋转可以用如下的迭代方程组来表示:

$$\begin{cases} x_{i+1} = (x_i - \sigma_i y_i \tan \theta_i) \cos \theta_i = (x_i - \sigma_i y_i 2^{-i}) \cos \theta_i \\ y_{i+1} = (y_i + \sigma_i x_i \tan \theta_i) \cos \theta_i = (y_i + \sigma_i x_i 2^{-i}) \cos \theta_i \\ z_{i+1} = z_i - \sigma_i \theta_i = z_i - \sigma_i \arctan(2^{-i}) \end{cases} \quad (3)$$

式中, z_i 为第 i 次旋转时的向量与目标向量之间的

误差角度, 即 $z_i = \theta - \sum_{j=0}^{i-1} \sigma_j \theta_j = \theta -$

$\sum_{j=0}^{i-1} \sigma_j \arctan(2^{-j})$, 它决定了第 i 次向量的旋转方向,

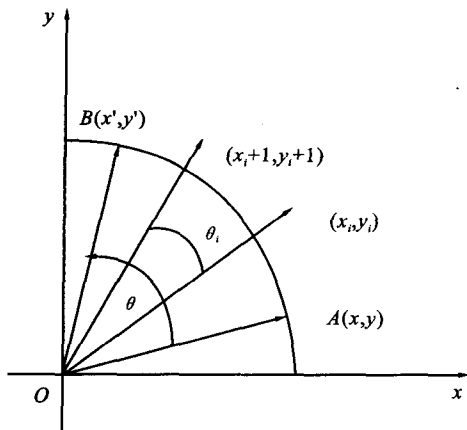


图2 CORDIC 算法迭代运算的原理图

并定义 $z_0 = \theta$; σ_i 为每次的旋转方向, $\sigma_i = \text{sign}(z_i)$; $\cos \theta_i$ 为每次迭代过程中的模校正因子, 也就是每次旋转时向量的模长所发生的变化。

令 $K_n = \prod_{i=0}^{n-1} \cos \theta_i$ 为经过 n 步迭代后总的模校正因子, 它的极限为

$$\begin{aligned} \lim_{n \rightarrow +\infty} K_n &= \lim_{n \rightarrow +\infty} \prod_{i=0}^{n-1} \cos \theta_i \\ &= \lim_{n \rightarrow +\infty} \prod_{i=0}^{n-1} \cos[\arctan(2^{-i})] \\ &= \lim_{n \rightarrow +\infty} \prod_{i=0}^{n-1} \frac{1}{\sqrt{1+2^{-2i}}} \approx 0.6073 \quad (4) \end{aligned}$$

从式(4)可以看出, 在理想情况下总的模校正因子 K_n 是个常数。而在一般情况下当迭代次数 n 确定后 K_n 也就唯一确定了, 因此 $\cos \theta_i$ 的乘积可以在迭代过程中先不处理, 而是被看作整个系统增益的一部分, 这样只需通过简单的移位和加减运算就能完成向量的旋转, 此时式(3)可以简化为

$$\begin{cases} x_{i+1} = x_i - \sigma_i y_i 2^{-i} \\ y_{i+1} = y_i + \sigma_i x_i 2^{-i} \\ z_{i+1} = z_i - \sigma_i \arctan 2^{-i} \end{cases}, \quad i = 0, 1, 2, \dots, n-1 \quad (5)$$

令旋转之后的增益为 $A_n = 1/K_n = \prod_{i=0}^{n-1} \sqrt{1+2^{-2i}}$, A_n 的精度与迭代次数 n 有关, 它的极限为

$$\lim_{n \rightarrow +\infty} A_n = \lim_{n \rightarrow +\infty} \prod_{i=0}^{n-1} \sqrt{1+2^{-2i}} \approx 1.647 \quad (6)$$

设初始向量 $\overline{OA}$ 的终点坐标为 (x_0, y_0) , 根据式(5)经过 n 步迭代后, 最终的结果为

$$\begin{cases} x_n = A_n(x_0 \cos z_0 - y_0 \sin z_0) \\ y_n = A_n(y_0 \cos z_0 + x_0 \sin z_0) \\ z_n \rightarrow 0 \end{cases} \quad (7)$$

式中, $z_n \rightarrow 0$ 表示向量通过不断地旋转最终与目标向量基本重合, 其重合程度取决于迭代次数 n 的大小。

由于 $\lim_{n \rightarrow +\infty} \sum_{i=0}^{n-1} \arctan(2^{-i}) \approx 99.88^\circ$, 所以 CORDIC 算法可以计算的旋转角度 $\theta \in [-99.88^\circ, 99.88^\circ]$ 。

3 改进型 CORDIC 算法

3.1 查找表的移位实现

在 CORDIC 算法中利用查找表来得到 $\arctan(2^{-i})$ 的离散值有 2 个劣势, 其一, 查找表需要

占用大量的ROM资源($2^n \times n\text{bit}$),并且随着 n 的增加,ROM的容量会呈现指数式的增长,这样就会占用更多的硬件资源;其二,CORDIC算法至少需要 n 级流水线,并且每级迭代都需要查找表,这样就不利于系统流水线结构的设计,容易形成速度“瓶颈”,从而降低了系统的运算性能.为了节省硬件资源,提高数据的处理速度,需要避开查找表.把 $\arctan(2^{-i})$ 用麦克劳林级数展开,可得

$$\arctan(2^{-i}) = 2^{-i} - \frac{2^{-3i}}{3} + \frac{2^{-5i}}{5} + O(2^{-7i}) = 2^{-i} - 2^{-3i-1} + \frac{2^{-3i}}{6} + \frac{2^{-5i}}{5} + O(2^{-7i}) \quad (8)$$

假设系统的位数为 n ,它与最终所需要的迭代次数相同,则系统的精度 $\epsilon = 2^{-n+1}$.当迭代到 m 次之后,为了能使 $\arctan(2^{-i}) \approx 2^{-i} - 2^{-3i-1}$,从而便于移位运算,则需要满足条件:

$$\xi = \sum_{i=m}^{n-1} \arctan(2^{-i}) - (2^{-i} - 2^{-3i-1}) \leq \epsilon = 2^{-n+1} \quad (9)$$

若取 $n = 16$,则 m 可以通过表1所示的数值分析来确定.

表1 m 取不同值时 ξ 的值

m	0	1	2	3	4	5
ξ	0.314	0.029	0.003	3.782e-4	4.669e-5	5.819e-6

由于 $2^{-15} \approx 3.0517\text{e}-5$,为了满足式(8),通过表1的数值分析可以得出 $m = 5$.这样当迭代次数从第5次开始,就可以利用 $2^{-i} - 2^{-3i-1}$ 来近似代替 $\arctan(2^{-i})$,从而避免了查找表的麻烦.

3.2 模校正因子的分解

从式(3)可以看出CORDIC算法中的向量旋转并非完美,因为旋转的过程会使向量的模长发生变化,这就需要用到模校正因子,该参数会给每一步的迭代带来相应的乘法运算,从而增加了硬件的实现难度,降低了数据的处理速度.为了在迭代中省去模校正因子,把 $\cos\theta_i$ 用麦克劳林级数展开,可得

$$\cos\theta_i = \cos[\arctan(2^{-i})] = \frac{1}{\sqrt{1+2^{-2i}}} = 1 - 2^{-2i-1} + O(2^{-4i-3}) \quad (10)$$

为了在式(3)中能用 $1 - 2^{-2i-1}$ 来近似代替 $\cos\theta_i$,从而便于移位运算,需要满足条件:

$$2^{-4m-3} < \epsilon = 2^{-n+1} \quad (11)$$

由于 $m = 5, n = 6, 2^{-4m-3} = 2^{-23} < 2^{-n+1} = 2^{-15}$,所以式(11)显然是成立的,这样就起到了省去模校正因子的效果.

把式(8)和式(10)中的近似表达式代入式(3),

改进型CORDIC算法中向量的旋转可以用如下的迭代方程组来表示:

$$\begin{cases} x_{i+1} = (1 - 2^{-2i-1})x_i - \sigma_i(2^{-i} - 2^{-3i-1})y_i \\ y_{i+1} = (1 - 2^{-2i-1})y_i + \sigma_i(2^{-i} - 2^{-3i-1})x_i \\ z_{i+1} = z_i - \sigma_i(2^{-i} - 2^{-3i-1}) \end{cases} \quad i = 5, 6, 7, \dots, n-1 \quad (12)$$

根据式(12)改进型CORDIC算法的流水线结构如图3所示(i 从相应的位置开始),图中ALU是加减运算单元, $>>$ 表示向右的移位运算.

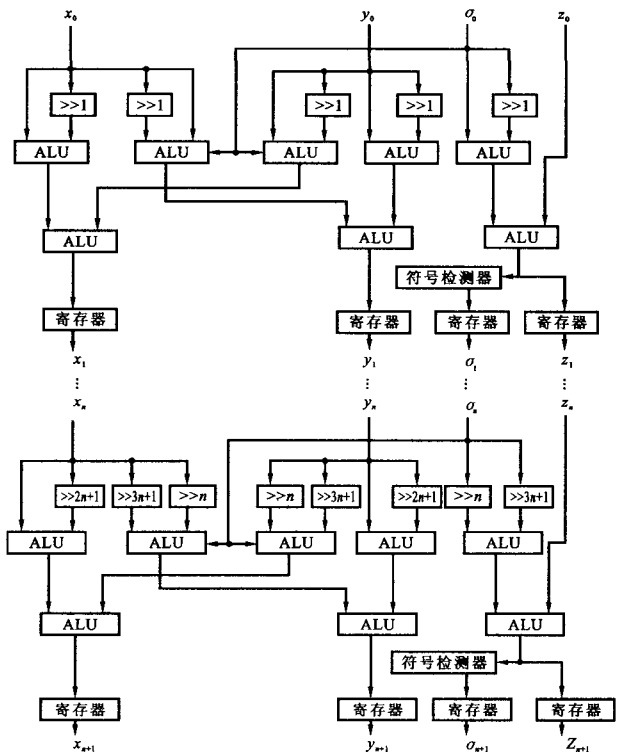


图3 改进型CORDIC算法的流水线结构

3.3 旋转角度的范围扩展

在改进型CORDIC算法中,旋转角度 $\theta \in [-\sum_{i=m}^{n-1} \arctan(2^{-i}), \sum_{i=m}^{n-1} \arctan(2^{-i})]$,当 $n = 16, m = 5$ 时, $\theta \in [-3.578^\circ, 3.578^\circ]$.为了能适用于所有的旋转角度,需要对旋转角度进行扩展.当 $n = 16$ 时,在第5次迭代中依次把 $i = 5$ 和 $i = 6$ 代入 $[-\sum_{i=m}^{n-1} \arctan(2^{-i}), \sum_{i=m}^{n-1} \arctan(2^{-i})]$ 重复迭代10次,即迭代序列为 $i = 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 7, 8, \dots, 14, 15$,则 $\theta \in [-30^\circ, 30^\circ]$.从理论上讲经过多次迭代可以使得 $\theta \in [-180^\circ, 180^\circ]$,但是重复迭代的次数过多会影响算法的运算性能,对此可以通过区域变换来实现.为了分析方便,假设初始向量 $\overrightarrow{OA}$ 的终点在圆上,且位于 x 轴的正半周,如图4所示.首先考虑将旋转角度 θ

由 $[-30^\circ, 30^\circ]$ 扩展到 $[-90^\circ, 90^\circ]$.

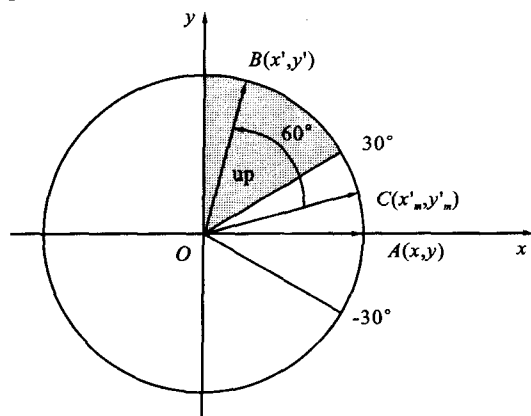


图4 旋转角度在 up 内的区域变换

若 θ 位于图4中的区域 up 内,其中 $\text{up} \in (30^\circ, 90^\circ]$.令 $\theta_m = \theta - 60^\circ$,则 $\theta_m \in (-30^\circ, 30^\circ]$, θ_m 就是初始向量 $\overrightarrow{OA}$ 旋转到向量 $\overrightarrow{OC}$ 的角度.根据式(1),向量 $\overrightarrow{OC}$ 逆时针旋转 60° 到目标向量 $\overrightarrow{OB}$ 可以表示为

$$\begin{cases} x' = x'_m \cos 60^\circ - y'_m \sin 60^\circ = 2^{-1}x'_m - \sqrt{3} \times 2^{-1}y'_m \\ y' = y'_m \cos 60^\circ + x'_m \sin 60^\circ = 2^{-1}y'_m + \sqrt{3} \times 2^{-1}x'_m \end{cases} \quad (13)$$

由于 $\theta_m \in (-30^\circ, 30^\circ]$,所以式(13)中的 (x'_m, y'_m) 可以通过改进型CORDIC算法求出.

若 $\theta \in [-30^\circ, 30^\circ]$,则 (x', y') 可以直接通过改进型CORDIC算法求出.

若 θ 位于图5中的区域 down 内,其中 $\text{down} \in [-90^\circ, -30^\circ]$.令 $\theta_m = \theta + 60^\circ$,则 $\theta_m \in [-30^\circ, 30^\circ]$.同理可得

$$\begin{cases} x' = 2^{-1}x'_m + \sqrt{3} \times 2^{-1}y'_m \\ y' = 2^{-1}y'_m - \sqrt{3} \times 2^{-1}x'_m \end{cases} \quad (14)$$

由于 $\theta_m \in [-30^\circ, 30^\circ]$,所以式(14)中的 (x'_m, y'_m) 可以通过改进型CORDIC算法求出.

利用同样的方法可以将改进型CORDIC算法中

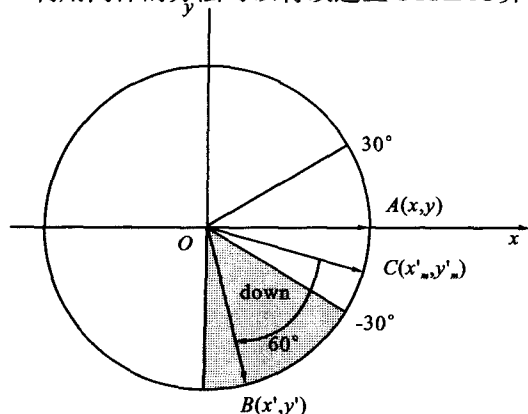


图5 旋转角度在 down 内的区域变换

的旋转角度 θ 由 $[-90^\circ, 90^\circ]$ 扩展到 $[180^\circ, 180^\circ]$.

综上所述,当旋转角度 $\theta \in [-90^\circ, 90^\circ]$ 时,改进型CORDIC算法的流程图如图6所示.

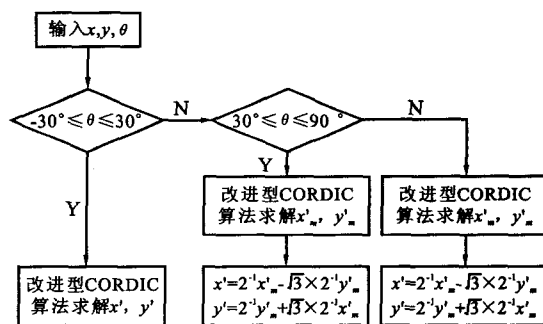


图6 $\theta \in [-90^\circ, 90^\circ]$ 时改进型CORDIC算法的流程图

3.4 改进型CORDIC算法的流水线结构分析

假设系统的位数 $n = 16$,改进型CORDIC算法和CORDIC算法的流水线结构比较如表2所示.

表2 两种算法的流水线结构比较

算法种类	流水线结构				
	选择器	乘法器	移位运算单元	加减运算单元	ROM存储单元
CORDIC算法	48	16	32	48	$2^{16} \times 16$
改进型CORDIC算法	32	0	128	96	0

从表2可以看出,虽然改进型CORDIC算法的流水线结构比CORDIC算法的流水线结构多用了96个移位运算单元以及48个加减运算单元,但是它节省了16个选择器并且没有乘法器以及ROM存储单元.考虑到移位运算单元和加减运算单元的电路结构比较简单,而选择器和乘法器的电路结构较为复杂,并且ROM存储单元的规模十分庞大.因此,与CORDIC算法的流水线结构相比,改进型CORDIC算法所带来的硬件资源优势还是非常明显的.

4 改进型CORDIC算法的误差分析

设旋转角度为 θ ,利用改进型CORDIC算法计算出的旋转角度为 θ_m ,则角度误差为 $\varepsilon = \theta_m - \theta$,它就是上述提到的系统精度,反映了系统的位数 n .将 θ_m 和 θ 分别代入式(1)可得

$$\begin{cases} x'_m = x \cos \theta_m - y \sin \theta_m \\ y'_m = y \cos \theta_m + x \sin \theta_m \end{cases} \quad (15)$$

$$\begin{cases} x' = x \cos \theta - y \sin \theta \\ y' = y \cos \theta + x \sin \theta \end{cases} \quad (16)$$

向量经过旋转后,根据式(15)和式(16)可以求

出目标向量的终点坐标对应的误差为

$$\begin{cases} x'_{\text{err}} = x'_m - x' = -2\sin 2^{-n} [x \sin \theta \cos 2^{-n} + \\ x \cos \theta \sin 2^{-n} + y \cos \theta \cos 2^{-n} - y \sin \theta \sin 2^{-n}] \\ y'_{\text{err}} = y'_m - y' = 2\sin 2^{-n} [x \cos \theta \cos 2^{-n} - \\ x \sin \theta \sin 2^{-n} - y \sin \theta \cos 2^{-n} - y \cos \theta \sin 2^{-n}] \end{cases} \quad (17)$$

设初始向量的终点坐标为 $(x, y) = (1, 1)$, 在matlab软件中利用改进型CORDIC算法得到目标向量的终点坐标对应的误差 x'_{err} 和 y'_{err} 随旋转角度 θ 和系统的位数 n 变化的曲线如图7和图8所示. 从图7和图8可以看出, 当系统的位数 n 较小时误差有一定的波动, 而在系统的位数 n 达到16以后, 对所有的旋转角度而言, 改进型CORDIC算法得到的误差已经非常小并且逐渐收敛几乎接近于0, 这说明改进型CORDIC算法具有非常优越的性能.

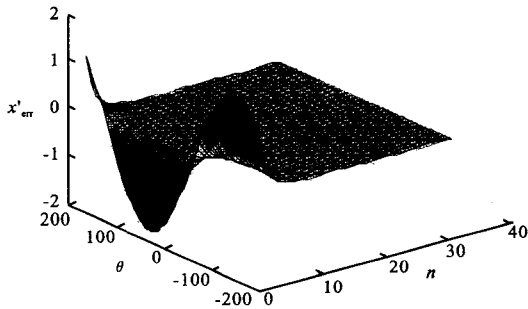


图7 x'_{err} 随 θ 和 n 变化的曲线

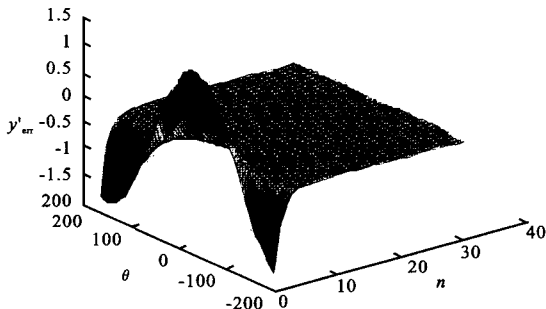


图8 y'_{err} 随 θ 和 n 变化的曲线

5 结束语

CORDIC算法在许多领域有着广泛的应用, 其迭代运算不仅需要利用查找表来计算离散化角度,

而且还需要利用乘法器来实现模校正因子的乘法运算, 此外它能够计算的角度范围也有限. 文中从查找表的移位实现、模校正因子的分解、旋转角度的范围扩展3个方面对CORDIC算法进行改进, 提出一种改进型CORDIC算法, 该算法的创新之处在于它不需要查找表和模校正因子, 硬件实现简单, 数据处理快速, 能够适用于所有的旋转角度并且具有很小的误差. 这为CORDIC算法在诸多领域的应用提供了一种新的思路.

参考文献:

- [1] Sumanasena M G B. A scale factor correction scheme for the CORDIC algorithm[J]. IEEE Transactions on Computers, 2008, 57(8): 1148 - 1152.
- [2] Granado J, Torralba A, Chávez J, et al. Optimization of CORDIC cells in the backward circular rotation mode[J]. AEU - International Journal of Electronics and Communications, 2007, 61(5): 337 - 340.
- [3] Wang H Z, Leray P, Palicot J. Reconfigurable architecture for MIMO systems based on CORDIC operators [J]. Comptes Rendus Physique, 2006, 7(7): 735 - 750.
- [4] Kebbaty H S, Blonde J P, Braun F. A new semiflat architecture for high speed and reduced area CORDIC chip[J]. Microelectronics Journal, 2006, 37(2): 181 - 187.
- [5] Juang T B. Low latency angle recoding methods for the higher bit - width parallel CORDIC rotator implementations[J]. IEEE Transactions on Circuits and Systems II: Express Briefs, 2008, 55(11): 1139 - 1143.
- [6] Sung T Y, Hsin H C. Design and simulation of reusable IP CORDIC core for special - purpose processors [J]. IET Computers & Digital Techniques, 2007, 1(5): 581 - 589.
- [7] Antelo E, Villalba J, Zapata E L. A low - latency pipelined 2D and 3D CORDIC processors[J]. IEEE Transactions on Computers, 2008, 57(3): 404 - 417.

作者简介:

张天瑜 男, (1980-), 硕士研究生, 讲师. 研究方向为高性能嵌入式处理器体系结构的设计、大规模集成电路系统的设计.