CORDIC

April 26, 2024

1 El algoritmo CORDIC

CORDIC (COordinate Rotation DIgital Computer) es un algoritmo de hardware iterativo que usa una sucesión de rotaciones para calcular una variedada de funciones elementales. Fue introducido por Volder [1] para el cáclulo de funciones trigonométricas y luego expandido por J. S. Walter [2], permitiendo calcular funciones lineales, logarítmicas, hiperbólicas, etc.

Sea $\mathbf{x} = (x, y)$ un vector del plano el cual es rotado por un ángulo ϕ. El vector rotado será $\mathbf{x}_\phi = (x_\phi, y_\phi)$ tal como se muestra en la Fig. 1.

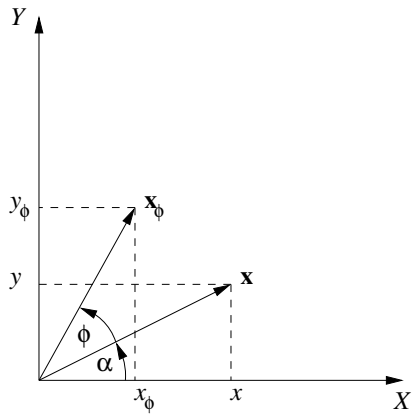

Figure 1: Rotación de un vector por un ángulo ϕ.

Si $\mathbf{x}$ es unitario, es decir $\|\mathbf{x}\| = 1$, entonces

$$\begin{aligned} x &= \cos(\alpha) \\ y &= \sin(\alpha) \end{aligned} \tag{1}$$

Recordemos las siguientes identidades trigonométricas:

$$\begin{aligned} \sin(\alpha + \beta) &= \sin(\alpha)\cos(\beta) + \cos(\alpha)\sin(\beta) \\ \cos(\alpha + \beta) &= \cos(\alpha)\cos(\beta) - \sin(\alpha)\sin(\beta) \end{aligned} \tag{2}$$

Resulta entonces:

$$\begin{aligned} x_\phi &= \cos(\alpha + \phi) = \cos(\alpha)\cos(\phi) - \sin(\alpha)\sin(\phi) = x\cos(\phi) - y\sin(\phi) \\ y_\phi &= \sin(\alpha + \phi) = \sin(\alpha)\cos(\phi) + \cos(\alpha)\sin(\phi) = y\cos(\phi) + x\sin(\phi) \end{aligned} \tag{3}$$

o equivalentemente en su forma matricial:

$$\begin{bmatrix} x_\phi \\ y_\phi \end{bmatrix} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{4}$$

Tomando como factor común $\cos(\phi)$ en la ec. (3) se obtiene:

$$\begin{aligned} x_\phi &= \cos(\phi)[x - y\tan(\phi)] \\ y_\phi &= \cos(\phi)[y + x\tan(\phi)] \end{aligned} \tag{5}$$

donde $\cos(\phi) > 0$ siempre que $-\pi/2 < \phi < \pi/2$.

La ecuación anterior muestra que es necesario realizar multiplicaciones y sumas/restas para una rotación. El algoritmo CORDIC se basa en dos ideas fundamentales para evitar las multiplicaciones. La primera es que una rotación por un ángulo arbitrario ϕ es equivalente a una sucesión de rotaciones parciales dadas por los ángulos $\{\phi_i\}, \quad i = 0, 1, 2, \ldots, n-1; \quad n \in \mathbb{N}$, provisto que $\phi = \sum_{i=0}^{n-1} \phi_i$, donde cada ϕ_i puede ser positivo o negativo. La segunda idea fundamental, es que cada ángulo elemental de rotación puede ser elegido según:

$$\tan(\phi_i) = 2^{-i}, \quad i = 0, 1, 2, \ldots, n-1; \quad n \in \mathbb{N} \tag{6}$$

In este caso, multiplicar por $\tan(\phi)$ se puede obtener por un shifting a derecha del vector binario.

$$\begin{aligned} x_{i+1} &= k_i \left[x_i - y_i d_i 2^{-i}\right] \\ y_{i+1} &= k_i \left[y_i + x_i d_i 2^{-i}\right] \end{aligned} \tag{7}$$

donde [1]

$$k_i = \cos(\phi_i) = \frac{1}{\sqrt{1 + 2^{-2i}}} \tag{8}$$

d_i determina el signo de cada rotación elemental, es decir d_i sólo puede tomar los valores 1 o -1. El algoritmo CORDIC siempre realiza una cantidad de rotaciones elementales con ángulos predefinidos y en cada paso de la iteración se determina si dicha rotación es a favor o en contra de las agujas del reloj asignado el valor ± 1 a d_i.

Dadas n rotaciones sucesivas sobre el vector $\mathbf{x_0} = [x_0 \ y_0]^T \in \mathbf{R}^2$, resulta:

$$\begin{aligned} x_n &= x_0 \cos\left(\sum_{i=0}^{n-1} d_i \arctan\left(2^{-i}\right)\right) - y_0 \sin\left(\sum_{i=0}^{n-1} d_i \arctan\left(2^{-i}\right)\right) \\ y_n &= x_0 \sin\left(\sum_{i=0}^{n-1} d_i \arctan\left(2^{-i}\right)\right) - y_0 \cos\left(\sum_{i=0}^{n-1} d_i \arctan\left(2^{-i}\right)\right) \end{aligned} \tag{9}$$

Luego de n rotaciones, el valor efectivo del ángulo de rotación será:

$$\phi_n = \sum_{i=0}^{n-1} d_i \arctan\left(2^{-i}\right) \tag{10}$$

Las siguientes definiciones son necesarias:

- El error $\epsilon_{\mathbf{x}}$ entre un valor x y una aproximación $\hat{x}$ está dado por su diferencia $\epsilon_x = x - \hat{x}$.

[1] La demostración de la ec. (8) se encuentra en el apéndice 7.

- El error absoluto e_x entre un valor x y una aproximación $\hat{x}$ es $e_x = |\epsilon_x| = |x - \hat{x}|$.
- La incertidumbre Δx del valor x es el mínimo de todas las cotas superiores M tales que $|x - \hat{x}| \leq M$.

Con esas definiciones, se puede escribir:

$$e_\phi = |\phi - \phi_n| \leq \frac{1}{2} \arctan\left(2^{-n+1}\right) = \Delta\phi \tag{11}$$

Se puede probar que [3]:

$$\lim_{n\to\infty} e_\phi = 0 \tag{12}$$

Fig. 2 muestra $\Delta\phi$ versus la cantidad de iteraciones n.

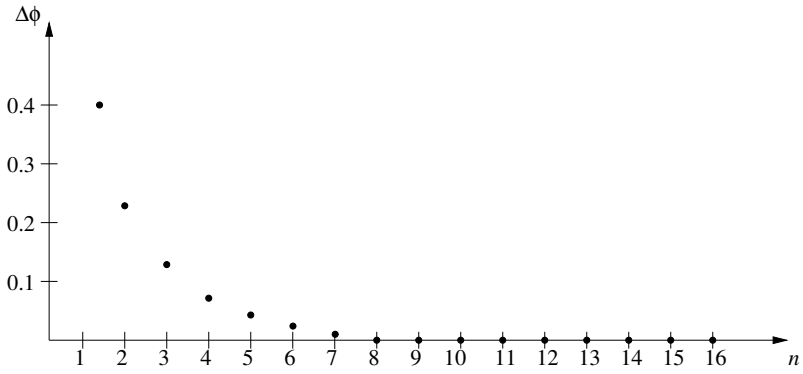

Figure 2: $\Delta\phi$ vs number of iterations n.

La multiplicación por k_i en cada paso de iteración actúa como una ganancia debido a que ambas componentes x and y son afectadas por la misma cantidad. Sin embargo el valor del ángulo final de rotación es el correcto.

Una posibilidad es ignorar k_i en las ecuaciones, obteniéndose la ec. 13, y tener en cuenta el factor de ganancia la final de la iteración:

$$\begin{aligned} x'_{i+1} &= x'_i - y'_i d_i 2^{-i} \\ y'_{i+1} &= y'_i + x'_i d_i 2^{-i} \end{aligned} \tag{13}$$

donde $i = 0, 1, \ldots, n-1$.
Se observa de la ec. (7) que:

$$
\begin{aligned}
x_{i+1} &= k_i \left(x_i - y_i d_i 2^{-i}\right) \\
&= k_i \left[k_{i-1}\left(x_{i-1} - y_{i-1} d_{i-1} 2^{-i+1}\right) - k_{i-1}\left(y_{i-1} + x_{i-1} d_{i-1} 2^{-i+1}\right) d_i 2^{-i}\right] \\
&= k_i k_{i-1}\left(x'_i - y'_i d'_i 2^{-i}\right) \\
\\
y_{i+1} &= k_i \left(y_i + x_i d_i 2^{-i}\right) \\
&= k_i \left[k_{i-1}\left(y_{i-1} + x_{i-1} d_{i-1} 2^{-i+1}\right) + k_{i-1}\left(x_{i-1} - y_{i-1} d_{i-1} 2^{-i+1}\right) d_i 2^{-i}\right] \\
&= k_i k_{i-1}\left(y'_i + x'_i d'_i 2^{-i}\right)
\end{aligned}
\tag{14}
$$

y entonces

$$
\begin{aligned}
x_n &= \left(\prod_{i=0}^{n-1} k_i\right) x'_n \\
y_n &= \left(\prod_{i=0}^{n-1} k_i\right) y'_n
\end{aligned}
\tag{15}
$$

Se concluye entonces que es más conveniente iterar por medio de la ec. (13), obteniendo x'_n and y'_n. Posteriormente, los valores x_n and y_n pueden hallarse postmultiplicando x'_n y y'_n por k_n,

$$
\begin{aligned}
x_n &= k_n x'_n \\
y_n &= k_n y'_n
\end{aligned}
\tag{16}
$$

donde

$$
k_n = \prod_{i=0}^{n-1} k_i = \prod_{i=0}^{n-1} \frac{1}{\sqrt{1 + 2^{-2i}}} \tag{17}
$$

Observa que a medida que las iteraciones avanzan, el ángulo de rotación se vuelve más pequeño, y entonces $\cos(\phi_i)$ se acerca cada vez más a 1 y su contribución a k_n se va volviendo insignificante.

$$
K = \lim_{n \to \infty} k_n \approx 0.607252935\,00888\,12561694 \tag{18}
$$

Por lo tanto, omitir el factor de escala, el conjunto de ecuaciones siguientes define el algoritmo CORDIC el cual provee un método de encontrar las coordenadas de un vector rotado x'_n, y'_n sin efectuar productos matriciales, tan sólo con el empleo de sumas, restas y shifting

$$
\begin{aligned}
x'_{i+1} &= x'_i - y'_i d_i 2^{-i} \\
y'_{i+1} &= y'_i + x'_i d_i 2^{-i}
\end{aligned}
\tag{19}
$$

Los valores x'_n, y'_n corresponden a las coordenadas del vector $\mathbf{x}$ rotado por un ángulo ϕ_n y escalado por una ganancia $A_n = 1/k_n$ conocida como CORDIC gain.

Fig. 3 muestra el efecto de estiramiento del vector en cada iteración. Generalmente cada iteración del algoritmo se conoce como pseudorotación debido a que no preserva la norma.

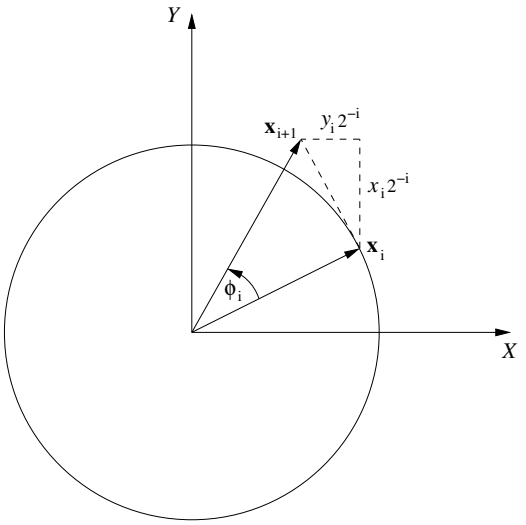

Figure 3: Pseudorotación producida por la iteración i.

El valor d_i en cada iteración es obtenido mediante la acumulación de las rotaciones previas y comparando la rotación obtenida en la iteración i con respecto al ángulo deseado. Si el ángulo deseado es más mayor (o menor) que la rotación previamente alcanzada, entonces este rota contrario (o a favor) del reloj en la próxima iteración. El procedimiento descripto es una suerte de un mecanismo de realimentación el cual integra las rotaciones realizadas, compara con el valor deseado y elige el sentido de la nueva rotación de forma tal de minimizar el error $\phi_{error} = \phi - \sum_{i=0}^{n-1} \phi_i$.

Este mecanismo de realimentación es implementado mediante el agregado de una tercera ecuación al al conjunto de ecuaciones del algoritmo CORDIC:

$$\begin{aligned} x'_{i+1} &= x'_i - y'_i d_i 2^{-i} \\ y'_{i+1} &= y'_i + x'_i d_i 2^{-i} \\ z_{i+1} &= z_i - d_i \arctan(2^{-i}) \end{aligned} \tag{20}$$

Esta última ecuación acumula el ángulo de las rotaciones previas y compara con el valor valor inicial $z[0]$, el cual corresponde al ángulo de rotación deseado:

$$d_i = \begin{cases} -1 & \text{if } z_i < 0, \\ +1 & \text{if } z_i \geq 0. \end{cases} \tag{21}$$

El cálculo de z sólo puede llevarse a cabo conociendo previamente los valores de las rotaciones elementales $\phi_i = \arctan\left(2^{-i}\right)$, $i = 0, 1, \ldots, n-1$. Estos valores pueden ser almacenados previamente en una lookup table.

2 Convergencia del algoritmo CORDIC

Para asegurar la convergencia del algoritmo luego de infinitas iteraciones debe asegurarse que el vector resultante converja en fase, es decir el ángulo ϕ_n se aproxima tanto como se desee al ángulo ϕ, y además la que converja en norma, lo que implica que el factor de escala A_n debe ser finito cuando n tiende a infinito.

La ecuación (11) muestra la convergencia del algoritmo respecto del ángulo de rotación debido a la monotonicidad creciente de la función arcotangente y la monotonicidad decreciente de su argumento 2^{-N+1}.

2.1 Convergencia en norma

Luego de n iteraciones, ambas salidas del algoritmo CORDIC se ven escaladas por un factor

$$A_n = \prod_{i=0}^{n-1} \sqrt{1 + 2^{-2i}} = \sqrt{\prod_{i=0}^{n-1} (1 + 2^{-2i})} \tag{22}$$

Para probar la convergencia de A_n, alcanza encontrar una cota superior y una cota inferior. Una cota inferior es simple de encontrar, ya que A_n corresponde a la productoria de factores todos ellos positivos independientemente del valor de i, por lo tanto $A_n \geq 0$, $\forall n \in \mathbf{N}$.

Para encontrar una cota superior se analizan los términos parciales de la sucesión $\left\{\prod_{i=0}^{n-1} \left(1 + 2^{-2i}\right)\right\}_{n \in \mathbf{N}}$.

1. $n = 0$

$$\prod_{i=0}^{n-1} 1 + 2^{-2i} = 1 + 2^0 = 2$$

2. $n = 1$

$$\prod_{i=0}^{n-1} 1 + 2^{-2i} = 2\left(1 + 2^{-2}\right) = 2 + 2^{-1}$$

3. $n = 2$

$$\prod_{i=0}^{n-1} 1 + 2^{-2i} = \left(2 + 2^{-1}\right)\left(1 + 2^{-4}\right) = 2 + 2^{-1} + 2^{-3} + 2^{-5}$$

4. $n = 3$

$$\begin{aligned} \prod_{i=0}^{n-1} 1 + 2^{-2i} &= \left(2 + 2^{-1} + 2^{-3} + 2^{-5}\right)\left(1 + 2^{-6}\right) \\ &= 2 + 2^{-1} + 2^{-3} + 2^{-4} + 2^{-7} + 2^{-9} + 2^{-11} \end{aligned}$$

5. $n = 4$

$$\begin{aligned}\prod_{i=0}^{n-1} 1 + 2^{-2i} &= \left(2 + 2^{-1} + 2^{-3} + 2^{-4} + 2^{-7} + 2^{-9} + 2^{-11}\right)\left(1 + 2^{-8}\right) \\ &= 2 + 2^{-1} + 2^{-3} + 2^{-4} + 2^{-6} + 2^{-8} + 2^{-10} + 2^{-12} + 2^{-15} \\ &\quad +2^{-17} + 2^{-19}\end{aligned}$$

6. $n = 5$

$$\begin{aligned}\prod_{i=0}^{n-1} 1 + 2^{-2i} &= (2 + 2^{-1} + 2^{-3} + 2^{-4} + 2^{-6} + 2^{-8} + 2^{-10} + 2^{-12} + 2^{-15} \\ &\quad +2^{-17} + 2^{-19})\left(1 + 2^{-10}\right) \\ &= 2 + 2^{-1} + 2^{-3} + 2^{-4} + 2^{-6} + 2^{-8} + 2^{-9} + 2^{-10} + 2^{-11} \\ &\quad +2^{-12} + 2^{-14} + 2^{-15} + 2^{-16} + 2^{-17} + 2^{-18} + 2^{-19} \\ &\quad +2^{-20} + 2^{-22} + 2^{-25} + 2^{-27} + 2^{-29}\end{aligned}$$

Se observa que el término de mayor orden corresponde a $2^{-\left(n^2+n-1\right)}$. Si bien no están presentes todos los términos hasta $2^{-\left(n^2+n-1\right)}$, se puede ver que

$$\prod_{i=0}^{n-1} 1 + 2^{-2i} < 1 + \sum_{i=0}^{k} 2^{-i} < 1 + \sum_{i=0}^{\infty} 2^{-i} = 3, \quad k = n^2 + n - 1 \tag{23}$$

Por lo tanto,

$$A_n = \prod_{i=0}^{n-1} \sqrt{1 + 2^{-2i}} < \sqrt{3} \cong 1.73205 \tag{24}$$

De esta forma se concluye que el algoritmo CORDIC converge luego de infinitas iteraciones alcanzando un error nulo en el ángulo de rotación y un factor de escala $A_n \cong 1,64676$.

Si se desean los valores de x_n y y_n sin el factor de escala, se puede multiplicar la salida del algoritmo, x'_n, y'_n por $k_n = 1/A_n$, el cual adquiere un valor de aproximadamente 0.60725 logrando así que el algoritmo conserve la norma original del vector de entrada $\mathbf{x}_0$.

Estas observaciones son formalizadas a continuación. A partir de (17) se define

$$k_n^2 = \prod_{i=0}^{n-1} \frac{1}{1 + 2^{-2i}} \tag{25}$$

La suceción $\left\{k_n^2\right\}_{n\in\mathbf{N}}$ es una sucesión decrecientes de números reales positivos.

Demostración:

Para demostrar que cada término de la sucesión es positivo basta observar que cada factor de la productoria definida en (25) es positivo y para demostrar que es decreciente basta observar a su vez que cada factor es menor a 1.

Estas dos observaciones implica que la sucesión $\left\{k_n^2\right\}_{n\in\mathbf{N}}$ está acotada inferiormente con lo cual es convergente y entonces se puede enunciar el siguiente teorema.

La suceción $\left\{k_n^2\right\}_{n\in\mathbf{N}}$ converge en $\mathbf{R}$. Aún más, converge a un número mayor a cero.

Demostración:

Al ser una sucesión monótonamente decreciente y acotada inferiormente por cero necesariamente converge por la completitud de $\mathbf{R}$.

La convergencia a un número mayor a cero se hará por el absurdo. Sunpóngase que efectivamente $\left\{k_n^2\right\}_{n\in\mathbf{N}}$ converge a 0. Entonces por definición de convergencia, para todo $\varepsilon > 0$ existe $N \in \mathbf{N}$ tal que si $n \geq N$ se cumple $|k_n^2 - 0| = k_n^2 < \varepsilon$. Por lo tanto,

$$\prod_{i=0}^{n-1} \frac{1}{1+2^{-2i}} < \varepsilon \tag{26}$$

Lo cual se cumple si y sólo si

$$1 < \varepsilon \prod_{i=0}^{n-1} 1 + 2^{-2i} \qquad \forall \varepsilon > 0 \tag{27}$$

Sea $0 < \varepsilon = 1/(2\prod_{i=0}^{n-1} 1 + 2^{-2i})$ entonces resulta de la ecuación 27 que $1 < 1/2$ lo cual es absurdo y parte de suponer que $\left\{k_n^2\right\}_{n\in\mathbf{N}}$ converge a 0.

De esta forma se concluye que si $\left\{k_n^2\right\}_{n\in\mathbf{N}}$ converge entonces también lo hace $\{k_n\}_{n\in\mathbf{N}}$ y en consecuencia $\{A_n\}_{n\in\mathbf{N}}$. Por lo que queda demostrada la convergencia en norma del algoritmo.

2.2 Convergencia en fase

La convergencia en fase se demuestra haciendo uso del llamado Teorema de Convergencia:

Sea $\{\sigma_0, \sigma_1, \ldots, \sigma_n\}$ una secesión finita decreciente de $n+1$ números positivos, es decir $\sigma_i \geq \sigma_j$ para $i \geq j$, la cual cumple que

$$\sigma_k \leq \sigma_n + \sum_{j=k+1}^{n} \sigma_j, \quad 0 \leq k \leq n \tag{28}$$

Además, sea r un número positvo que satisface

$$|r| \leq \sum_{j=0}^{n} \sigma_j \tag{29}$$

Sea también la sucesión $s_0 = 0$, $s_{k+1} = s_k + \delta_k \sigma_k$, $k = 0, 1, \ldots, n$, donde

$$\delta_k = sgn(r - s_k) = \begin{cases} 1, & r \geq s_k \\ -1, & r \leq s_k \end{cases} \tag{30}$$

Entonces,

$$|r - s_k| \leq \sigma_n + \sum_{j=k}^{n} \sigma_j, \quad 0 \leq k \leq n \tag{31}$$

En particular,

$$|r - s_{n+1}| \leq \sigma_n \tag{32}$$

Demostración:
La demostración se realizará por inducción sobre k. Para $k = 0$, se tiene

$$|r - s_0| = |r| \leq \sigma_n + \sum_{j=0}^{n} \sigma_j$$

Asumiendo el teorema válido para k, se tiene la siguiente hipótesis inductiva (H1)

$$|r - s_k| \leq \sigma_n + \sum_{j=k+1}^{n} \sigma_j$$

Considerando el caso $k+1$, se debe demostrar que la siguiente tesis inductiva (T1) es válida

$$|r - s_{k+1}| \leq \sigma_n + \sum_{j=k+1}^{n} \sigma_j$$

Considérese $|r - s_{k+1}| = |r - s_k - \delta_k \sigma_k|$. Si $r - s_k \geq 0$, entonces $\delta_k = 1$ y $|r - s_k - \delta_k \sigma_k| = ||r - s_k| - \sigma_k|$. Por el contrario, si $r - s_k < 0$, entonces $\delta_k = -1$ y $|r - s_k - \delta_k \sigma_k| = |r - s_k + \sigma_k| = ||r - s_k| - \sigma_k|$. Por lo tanto, en ambos casos

$$|r - s_{k+1}| = ||r - s_k| - \sigma_k| \leq \sigma_n + \sum_{j=k+1}^{n} \sigma_j$$

Lo que muestra que la tésis inductiva T1 es válida. Finalmente, $-\sigma_n \leq \|r - s_n\| - \sigma_n \leq 2\sigma_n - \sigma_n = \sigma_n$ y entonces $|r - s_{n+1}| = ||r - s_n| - \sigma_n| \leq \sigma_n$, lo cual completa la demostración.

Para $n > 3$, la sucesión $\sigma_k = \arctan\left(2^{-k}\right)$, $k = 0, 1, \ldots, n$ satisface las hipótesis del teorema de convergencia para todo $|r| \leq \pi/2$.
Demostración:
La secuencia $\arctan\left(2^0\right), \arctan\left(2^{-1}\right), \ldots, \arctan\left(2^{-n}\right) = \arctan(1), \ldots, \arctan\left(1/2^n\right)$ es claramente una secuencia decreciente de valores positivos. El Teorema del Valor Medio establece que existe un número c entre a y b, $a < c < b$, tal que

$$\frac{\arctan(b) - \arctan(a)}{b-a} = \frac{1}{1+c^2} \tag{33}$$

Sean $a = 2^{-(j+1)}$, $b = 2^{-j}$ en (33). Entonces, $b - a = 2^{-(j+1)}$ y además

$$\frac{1}{1+c^2} < \frac{1}{1+a^2} = \frac{1}{1+2^{-2(j+1)}} = \frac{2^{2j+2}}{1+2^{2j+2}} \tag{34}$$

Entonces,

$$\sigma_j - \sigma_{j+1} < (b-a)\frac{1}{1+c^2} \leq \frac{1}{2^{j+1}}\frac{2^{2j+2}}{1+2^{2j+2}} = \frac{2^{j+1}}{1+2^{2j+1}} \tag{35}$$

Sean ahora $a = 0$, $b = 2^{-j}$ en (33). Entonces,

$$\frac{1}{1+c^2} > \frac{1}{1+b^2} = \frac{1}{1+2^{-2j}} = \frac{2^{2j}}{1+2^{2j}} \tag{36}$$

y

$$\sigma_j = b\frac{1}{1+c^2} \geq \frac{1}{2^j}\frac{2^{2j}}{1+2^{2j}} = \frac{2^j}{1+2^{2j}} \tag{37}$$

Por otra parte,

$$\sigma_k - \sigma_n = (\sigma_k - \sigma_{k+1}) + (\sigma_{k+1} - \sigma_{k+2}) + \ldots + (\sigma_{n-1} - \sigma_n) = \sum_{j=k}^{n-1} \sigma_j - \sigma_{j+1} \tag{38}$$

Utilizando (35) en (38),

$$\sigma_k - \sigma_n \leq \sum_{j=k}^{n-1} \frac{2^{j+1}}{1+2^{2j+2}} = \sum_{j=k+1}^{n} \frac{2^j}{1+2^{2j}} \leq \sum_{j=k+1}^{n} \sigma_j \tag{39}$$

con lo cual se concluye que

$$\sigma_k \leq \sigma_n + \sum_{j=k+1}^{n} \sigma_j, \quad \forall\, 0 \leq k \leq n \tag{40}$$

Por último, como la suma de los términos arctan(1), arctan(1/2), arctan(1/4) arctan(1/8) es mayor a $\pi/2$, entonces se concluye que

$$|r| \leq \frac{\pi}{2} < \sum_{j=0}^{3} \arctan\left(2^{-j}\right) < \sigma_n + \sum_{j=0}^{n} \sigma_j \tag{41}$$

lo que completa la demostración.

El algoritmo CORDIC converge en fase.
Demostración:

Sea la secuancia $s_k = \phi - z_k = \sum_{j=0}^{k-1} d_j \sigma_j$. Se observa que $s_0 = \phi - z_0 = 0$ y $s_{k+1} = \sum_{j=0}^{k} d_j \sigma_j = s_k + d_k \sigma_k$. Para $r = \phi$ se tiene $\rho_k = \text{signo}(r - s_k) = \text{signo}(\phi - s_k) = \text{signo}(z_k) = d_k$. Por lo tanto, la secuancia

$$|\phi - s_{n+1}| \leq \sigma_n = \arctan\left(2^{-n}\right) \leq \arctan\left(\frac{1}{2^n}\right) \tag{42}$$

satisface el Teorema de Convergencia, lo cual demuestra que el algoritmo CORDIC converge para cualquier secuencia $\{d_i\}_{i=0,\ldots,n-1}$.

3 Modos de operación

El algoritmo CORDIC puede operar en dos modos diferentes: rotación y vectorización.

3.1 Modo rotación

En modo rotación, el algoritmo CORDIC rota un vector inicial por un ángulo ϕ. Los datos de entrada corresponden a las coordenadas iniciales x_0, y_0 del vector y el ángulo deseado de rotación z_0. Luego de n iteraciones, el acumulador de fase será cercano a cero. Por lo tanto, el algoritmo CORDIC en modo rotación está definido por el siguiente conjunto de ecuaciones:

$$\begin{cases} x_{i+1} = x_i - y_i d_i 2^{-i} \\ y_{i+1} = y_i + x_i d_i 2^{-i} \\ z_{i+1} = z_i - d_i \arctan\left(2^{-i}\right) \end{cases} \qquad i = 0, 1, \ldots, n-1 \tag{43}$$

donde

$$\begin{aligned} d_i &= \begin{cases} -1 & \text{if } z_i < 0, \\ +1 & \text{if } z_i \geq 0. \end{cases} \\ x_0 &= x \\ y_0 &= y \\ z_0 &= \phi \end{aligned} \tag{44}$$

La salida del algoritmo luego de n iterations es la siguiente:

$$\begin{aligned} x_n &= A_n \left[x_0 \cos(\phi_n) - y_0 \sin(\phi_n)\right] \\ y_n &= A_n \left[y_0 \cos(\phi_n) + x_0 \sin(\phi_n)\right] \\ |z_n| &\leq \frac{1}{2} \arctan\left(2^{-n+1}\right) \\ A_n &= \prod_{i=0}^{n-1} \sqrt{1 + 2^{-2i}} \end{aligned} \tag{45}$$

donde el valor real del ángulo de rotación $\phi_n = z_n$ está dado por la ec. (10) y está cercano al valor deseado de rotación $\phi = z_0$ tal como indicado por la ec. (11).

Cuando n tiende a infinito, resulta:

$$\begin{aligned} x_n &= A_n \left[x_0 \cos(\phi) - y_0 \sin(\phi)\right] \\ y_n &= A_n \left[y_0 \cos(\phi) + x_0 \sin(\phi)\right] \\ z_n &= 0 \end{aligned} \tag{46}$$

$$A_{n-1} = 1,646765\ldots$$

3.2 Modo vectorización

En modo vectorización, el vector de entrada es rotado en dirección al eje x, de forma tal que el acumulador de fase finaliza con el valor inicial del ángulo del vector de entrada. En este modo de operación, el algoritmo reduce sucesivamente el valor absoluto de la coordenada y y entonces cuando $y = 0$, el vector queda alineado con el eje x. En modo vectorización, el algoritmo CORDIC se define por el siguiente conjunto de ecuaciones:

$$\begin{cases} x_{i+1} = x_i - y_i d_i 2^{-i} \\ y_{i+1} = y_i + x_i d_i 2^{-i} \\ z_{i+1} = z_i - d_i \arctan\left(2^{-i}\right) \end{cases} \qquad i = 0, 1, \ldots, n-1 \tag{47}$$

donde

$$\begin{aligned} d_i &= \begin{cases} -1 & \text{if } y_i \geq 0, \\ +1 & \text{if } y_i < 0. \end{cases} \\ x_0 &= x \\ y_0 &= y \\ z_0 &= \phi \end{aligned} \tag{48}$$

La salida del algoritmo luego de n iteraciones es la siguiente:

$$\begin{aligned} x_n &= A_n \sqrt{x_0^2 + y_0^2} \\ y_n &= 0 \\ z_n &= z_0 + \arctan\left(\frac{y_0}{x_0}\right) \end{aligned} \tag{49}$$

$$A_n = \prod_{i=0}^{n-1} \sqrt{1 + 2^{-2i}}$$

3.3 Extensión del rango

En el modo rotación, el algoritmo CORDIC trabaja de forma correcta siempre que el ángulo de rotación se encuentra entre $-\pi/2$ and $\pi/2$. Esto es debido a usar 2^0 para la primer iteración. Si z_0 está fuera de este rango, una rotación inicial por $\pi/2$ es realizada con el objetivo de ajustar la posición inicial de forma tal que el ángulo de rotación quede dentro del rango aceptado.

En la Fig. 4, if $z_0 > \pi/2$, una rotación $\pi/2$ inicial es realizada por lo cual los valores de entrada al algoritmo son los siguientes:

$$\begin{aligned} x'_0 &= -y_0 \\ y'_0 &= x_0 \\ z'_0 &= z_0 - \pi/2 \end{aligned} \tag{50}$$

Por otro lado, si $z_0 < \pi/2$, una rotación inicial de $-\pi/2$ es realizada y por lo tanto los valores de entrada al algoritmo son los siguientes:

$$\begin{aligned} x'_0 &= y_0 \\ y'_0 &= -x_0 \\ z'_0 &= z_0 + \pi/2 \end{aligned} \tag{51}$$

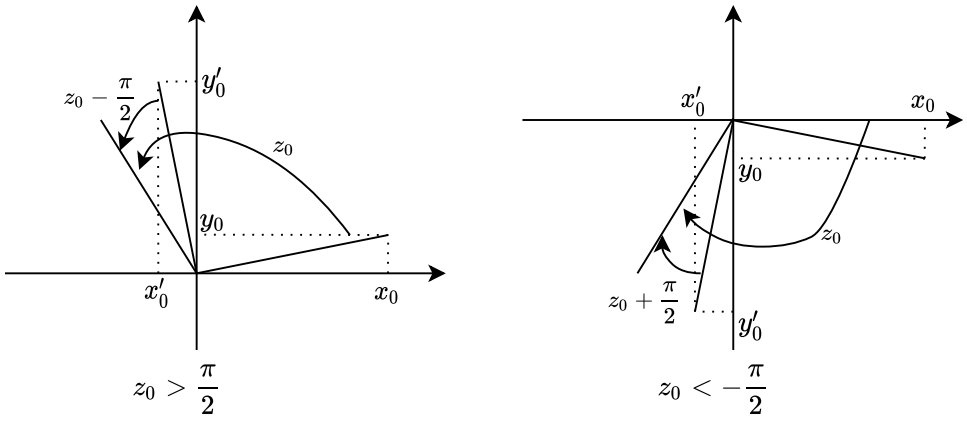

Figure 4: Extensión de rango para el modo rotación.

En el modo vectorización, el algoritmo CORDIC trabaja adecuadamente siempre que $x_0 > 0$. Si por el contrario $x_0 < 0$, una rotación inicial de $\pi/2$ es realizada con el objetivo de ajustar la posición inicial. El algoritmo CORDIC entonces realiza las iteraciones necesarias para llevar esta posición inicial ajustada en dirección al eje x.

Tal como se muestra en la Fig. 5, si $y_0 < 0$, la rotación inicial $\pi/2$ es realizada y los valores de entrada al algoritmo por lo tanto son:

$$\begin{aligned} x'_0 &= -y_0 \\ y'_0 &= x_0 \\ z'_0 &= z_0 - \pi/2 \end{aligned} \tag{52}$$

Por otro lado, si $y_0 > 0$, una rotación inicial $-\pi/2$ es realizada y los valores de entrada al algoritmo son:

$$\begin{aligned} x'_0 &= y_0 \\ y'_0 &= -x_0 \\ z'_0 &= z_0 + \pi/2 \end{aligned} \tag{53}$$

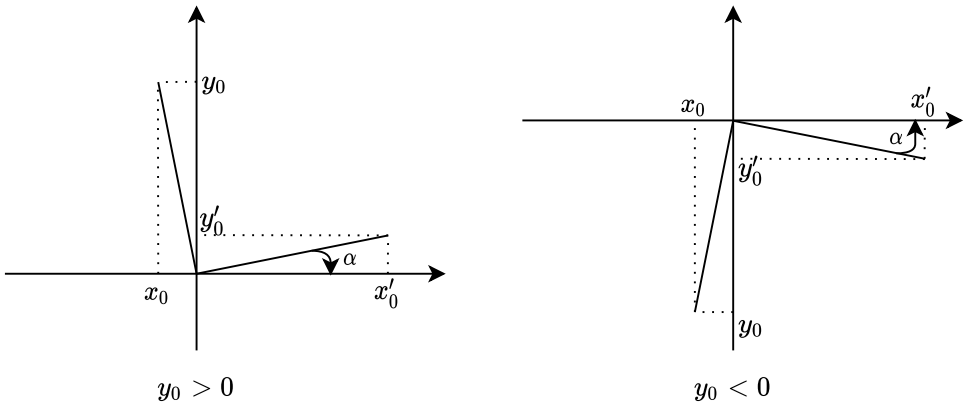

Figure 5: Extensión de rango para el modo vectorización.

Si $z_0 = 0$, el valor final de z_n es $\alpha + \pi/2$ or $\alpha - \pi/2$, el cual corresponde al ángulo total entre el eje x y la posición inicial.

4 Cálculo de funciones trigonométricas

El algoritmo CORDIC puede ser usado para computar diferentes funciones dependiendo del modo de operación (rotación o vectorización) y eligiendo inteligentemente los valores de entrada ($x0$, y_0, z_0).

4.1 Sin and cos

El algoritmo CORDIC puede ser utilizado para computar las funciones *sin* and *cos* del ángulo de entrada z_0 forzando $y_0 = 0$:

$$\begin{aligned} x_n &= A_n x_0 \cos(z_0) \\ y_n &= A_n x_0 \sin(z_0) \end{aligned} \tag{54}$$

Eligiendo $x_0 = 1/A_n$ entonces $x_n = \cos(z_0)$ y $y_n = \sin(z_0)$.

4.2 Conversión de coordenadas polares a cartesianas

Como se sabe, la conversion de coordenadas polares a cartesianas está dado por:

$$\begin{aligned} x &= r\cos(\phi) \\ y &= r\sin(\phi) \end{aligned} \tag{55}$$

Eligiendo $x_0 = r/A_n$, $y_0 = 0$ y $z_0 = \phi$, la salida del algoritmo CORDIC en modo rotación representa la conversión de coordenadas polares a cartesianas del vector de entrada.

4.3 Arctangent

La función arco tangente se computa mediante el algoritmo en modo vectorización. De la ec. (49), eligiendo $z_0 = 0$ se obtiene:

$$z_n = \arctan\left(\frac{y_0}{x_0}\right) \tag{56}$$

4.4 Norma

De la misma forma que la función arco tangente, la norma del vector de entrada se computa por el algorirmo operando en modo vectorización. De la ec. (49) se obtiene:

$$x_n = A_n\sqrt{x_0^2 + y_0^2} \tag{57}$$

Observar que el valor de salida se escala por la ganancia de CORDIC. En el caso de esto no ser aceptable, el valor verdadero de la norma del vector de entrada puede ser obtenido eligiendo como entradas $x_0 = x/A_n, y_0 = y/A_n$ o post-multiplicando la norma de salida x_n por $k_n = 1/A_n$.

4.5 Conversión de coordenadas cartesianas a polares

La conversión de coordenadas cartesianas a polares está definida por:

$$\begin{aligned} r &= \sqrt{x^2 + y^2} \\ \phi &= \arctan(y/x) \end{aligned} \tag{58}$$

De la ec. (49), la conversión cartesiana a polar se computa mediante el algoritmo CORDIC en modo vectorización eligiendo $z_0 = 0$, lo cual se muestra en la Fig. 6.

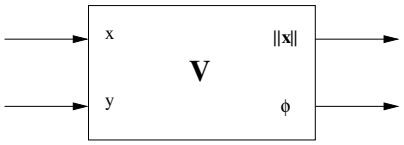

Figure 6: Conversión de coordenadas cartesianas a polares.

4.6 Rotación de un vector

La rotación de un vector (x, y) por un ángulo *phi* está definido por:

$$\begin{aligned} x_r &= x\cos(\phi) - y\sin(\phi) \\ y_r &= x\sin(\phi) + y\cos(\phi) \end{aligned} \tag{59}$$

De la ec. (46), la rotación del vector se computa mediante el algoritmo operando en modo rotación eligiendo $z_0 = \phi$,lo cual se muestra en la Fig. 7.

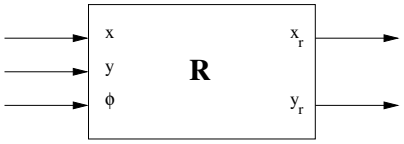

Figure 7: Rotación de un vector.

Notice again that the output values are scaled by the CORDIC gain. When this will not be acceptable, the true value of the rotated coordinates can be obtained choosing as inputs $x_0 = x/A_n, y_0 = y/A_n$ or by post-multiplying the outputs x_n, y_n by $k_n = 1/A_n$.

5 Implementación del algoritmo CORDIC

El algoritmo CORDIC puede ser implementado en aritmética de punto fijo o flotante. No hay cambios arquitecturales dependiendo del systema de representación numérica.

Teniendo precomputados los n valores de $atan(2^{-i})$, $i = 0, 1, \ldots, n-1$, entonces las ec. (43) y (47) se aplican dependiendo el modo de operación deseado.

En el caso de punto fijo, el complemento a dos (2C) se representa según:

$$x = \frac{1}{2^b}\left[-x_{N-1}2^{N-1} + \sum_{i=0}^{N-2} x_i 2^i\right] \tag{60}$$

donde N es el número total de bits y F es el número bits fraccionales. El rango numérico de representación es $-2^{N-1-F} \leq x \leq 2^{N-1-F} - \frac{1}{2^F}$. Notar que el producto por potencias de dos $2^{-i}x_i$ and $2^{-i}y_i$ se reducen a un shifting aritmético a derecha de x_i e y_i respectivamente, el cual es denotado según $2^{-i}x = x \gg i$.

En el caso de aritmética de punto flotante, un valor numérico se representa según:

$$x = m_x 2^{e_x} \tag{61}$$

donde m_x es la mantisa y e_x es el exponente, ambos signados. Entonces, $2^{-i}x = m_x.2^{e_x - i}$ lo cual reduce la operación de producto por potencia de dos a una resta, la cual también será denotada por $2^{-i}x = x \gg i$. Entonces,

dependiendo del contexto (punto fijo o flotante), el símbolo $\gg$ representa un shifting aritmético a derecha o una resta en el exponente.

Hay dos implementaciones principales del algoritmo: la implementación enrollada) y la desenrollada [4], las cuales se muestran en las Figs. 8 y 9. La arquitectura desenrollada puede ser pipelined por medio de registros entre etapas.

Para implementación más complejas pero eficientes ver [5].

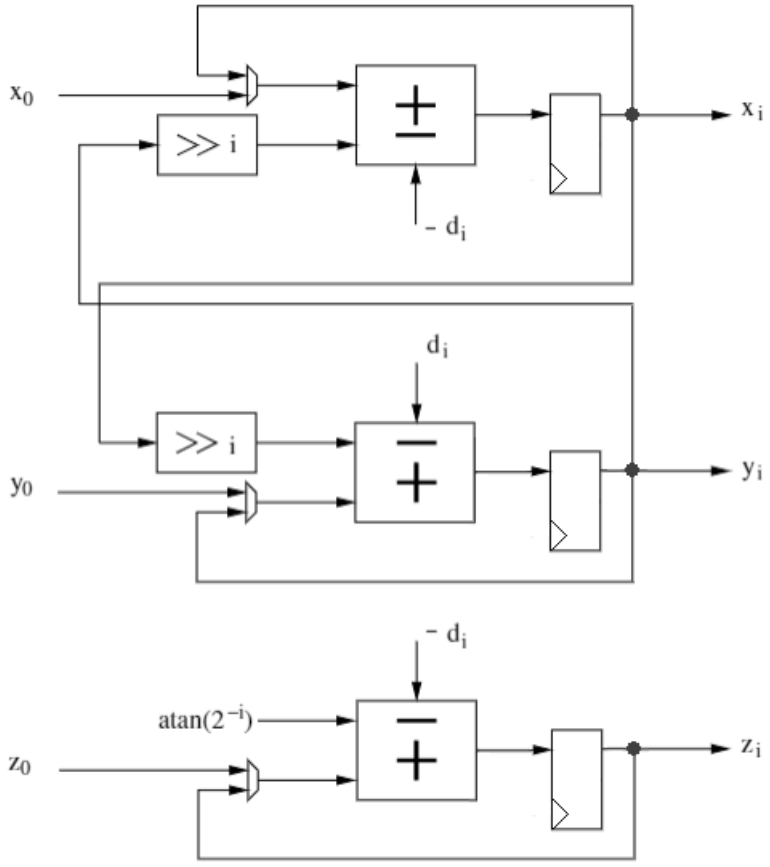

Figure 8: Arquitectura CORDIC enrollada.

6 CORDIC generalizado

El algoritmo CORDIC visto hasta el momento puede ser generalizado [2]. Se definen las ecuaciones de CORDIC generalizado según:

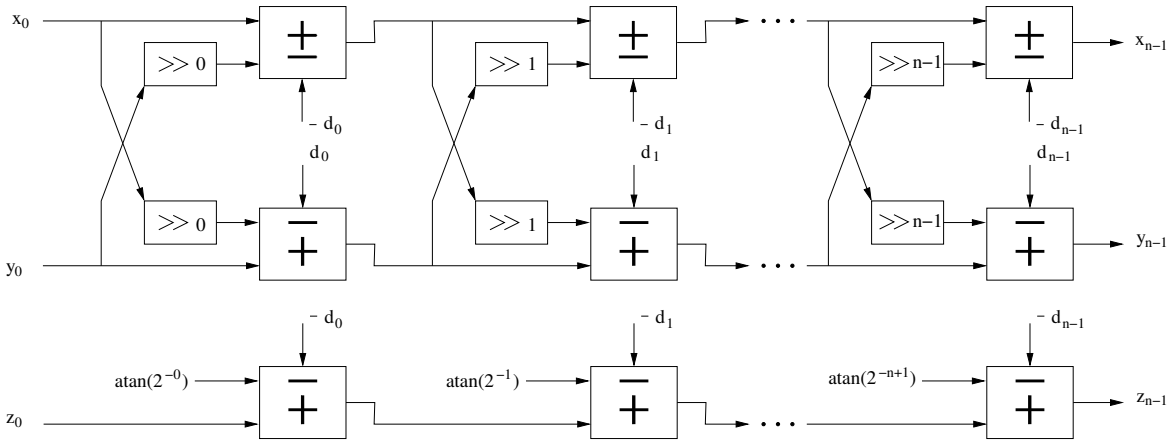

Figure 9: Arquitectura CORDIC desenrollada.

$$\begin{cases} x_{i+1} = x_i - \mu y_i d_i 2^{-i} \\ y_{i+1} = y_i + x_i d_i 2^{-i} \\ z_{i+1} = z_i - d_i f\left(2^{-i}\right) \end{cases} \quad i = 0, 1, \ldots, n-1 \tag{62}$$

donde:

- Rotación circular: $\mu = 1$, $f(x) = \arctan(x)$
- Rotación lineal: $\mu = 0$, $f(x) = x$
- Rotación hiperbólica: $\mu = -1$, $f(x) = \operatorname{arctanh}(x)$

6.1 Caso lineal

Para el caso lineal se toma $\mu = 0$y $f(x) = x$, obteniéndose:

$$\begin{aligned} x_{i+1} &= x_i \\ y_{i+1} &= y_i + x_i d_i 2^{-i} \\ z_{i+1} &= z_i - d_i 2^{-i} \end{aligned} \tag{63}$$

En el modo rotación, donde

$$d_i = \begin{cases} -1 & \text{if } z_i < 0 \\ +1 & \text{if } z_i \geq 0 \end{cases} \tag{64}$$

la rotación lineal produce:

$$\begin{aligned} x_n &= x_0 \\ y_n &= y_0 + x_0 z_0 \\ z_n &= 0 \end{aligned} \tag{65}$$

Mientras que en el modo vectorización, donde

$$d_i = \begin{cases} -1 & \text{if } y_i \geq 0 \\ +1 & \text{if } y_i < 0 \end{cases} \tag{66}$$

la rotación lineal produce:

$$\begin{aligned} x_n &= x_0 \\ y_n &= z_0 - y_0/x_0 \\ z_n &= 0 \end{aligned} \tag{67}$$

6.2 Caso hiperbólico

Para el caso hiperbólico se toma $\mu = -1$y $f(x) = \text{arctanh}(x)$, obteniéndose:

$$\begin{aligned} x_{i+1} &= x_i + y_i d_i 2^{-i} \\ y_{i+1} &= y_i + x_i d_i 2^{-i} \\ z_{i+1} &= z_i - d_i \text{arctanh}(2^{-i}) \end{aligned} \tag{68}$$

En el modo rotación, donde

$$d_i = \begin{cases} -1 & \text{if } z_i < 0 \\ +1 & \text{if } z_i \geq 0 \end{cases} \tag{69}$$

la rotación hiperbólica produce:

$$\begin{aligned} x_n &= A_n(x_0 \cosh(z_0) + y_0 \sinh(z_0)) \\ y_n &= A_n(y_0 \cosh(z_0) + x_0 \sinh(z_0)) \\ z_n &= 0 \\ A_n &= \textstyle\prod_{i=0}^{n-1} \sqrt{1 - 2^{-2i}} = 0.80\ldots \end{aligned} \tag{70}$$

Mientras que en el modo vectorización, donde

$$d_i = \begin{cases} -1 & \text{if } y_i \geq 0 \\ +1 & \text{if } y_i < 0 \end{cases} \tag{71}$$

la vectorización hiperbólica produce:

$$\begin{aligned} x_n &= A_n \sqrt{x_0^2 - y_0^2} \\ y_n &= 0 \\ z_n &= z_0 + \text{arctanh}\,(y_0/x_0) \\ A_n &= \textstyle\prod_{i=0}^{n-1} \sqrt{1 - 2^{-2i}} = 0.80\ldots \end{aligned} \tag{72}$$

Las rotaciones en el sistema hiperbólico no son convergentes. Sin embargo, en [2] se prueba que se pueden conseguir los resultados esperados repitiendo algunas de las iteraciones efectuadas. Dichas iteraciones a repetirse son las de índice $3k + 1$.

6.2.1 Cálculo de funciones elementales

Exponencial

Recordando que:

$$\begin{aligned} \cosh x &= \tfrac{e^x+e^{-x}}{2} \\ \sinh x &= \tfrac{e^x-e^{-x}}{2} \end{aligned} \tag{73}$$

Entonces,

$$\cosh x + \sinh x = e^x \tag{74}$$

Por lo cual, operando el CORDIC en rotación hiperbólica y haciendo que $x_0 = y_0 = 1/A_n$ se obtiene

$$x_n = y_n = e^{z_0} \tag{75}$$

Exponencial negativa

Recordando que:

$$\begin{aligned} \cosh x &= \tfrac{e^x+e^{-x}}{2} \\ \sinh x &= \tfrac{e^x-e^{-x}}{2} \\ \sinh -x &= \tfrac{e^{-x}-e^{x}}{2} = -\sinh x \end{aligned} \tag{76}$$

Entonces,

$$\cosh x + \sinh -x = \cosh x - \sinh x = e^{-x} \tag{77}$$

Por lo cual, operando el CORDIC en rotación hiperbólica y haciendo que $x_0 = -y_0 = 1/A_n$ se obtiene

$$x_n = y_n = e^{-z_0} \tag{78}$$

Logaritmo

Recordando que:

$$\ln x = 2 \operatorname{arctanh}\left(\frac{x-1}{x+1}\right) \tag{79}$$

Por lo cual, operando el CORDIC en vectorización hiperbólica y haciendo $x_0 = x+1$, $y_0 = x-1$, $z_0 = 0$, se obtiene

$$\begin{aligned} x_n &= 2A_n\sqrt{x} \\ y_n &= 0 \\ z_n &= \ln x \end{aligned} \tag{80}$$

Raiz cuadrada

Similar a la función logaritmo, se opera el CORDIC en modo rotación hiperbólica con entradas $x_0 = x/A_n^2 + 1/4$, $y_0 = x/A_n^2 - 1/4$, $z_0 = 0$ y se obtiene:

$$\begin{aligned} x_n &= \sqrt{x} \\ y_n &= 0 \\ z_n &= \ln(x/A_n^2) \end{aligned} \tag{81}$$

7 Apéndice. Demostración de la ec. (8)

Si $\phi \neq \pm\pi/2$, entonces:

$$\tan(\phi_i) = \frac{\sin(\phi_i)}{\cos(\phi_i)} \tag{82}$$

Restringiendo ϕ_i de forma ta lque sólo acepte aquellos valores para los cuales su tangente corresponde a potencias enteras negativas de dos, es decir $\tan(\phi_i) = 2^{-i}$ y elevando al cuadrado la ec. (82) se obtiene

$$\tan^2(\phi_i) = 2^{-2i} = \frac{\sin^2(\phi_i)}{\cos^2(\phi_i)} \tag{83}$$

Siendo $\sin^2(\phi_i) + \cos^2(\phi_i) = 1$, se cumple

$$2^{-2i} = \frac{1 - \cos^2(\phi_i)}{\cos^2(\phi_i)} = \frac{1}{\cos^2(\phi_i)} - 1 \tag{84}$$

y por lo tanto

$$1 + 2^{-2i} = \frac{1}{\cos^2(\phi_i)} \tag{85}$$

Finalmente,

$$k_i = \cos(\phi_i) = \frac{1}{\sqrt{1 + 2^{-2i}}} \tag{86}$$

References

[1] J. E. Volder, "The CORDIC Trigonometric Computing Technique", IRE Trans. on Electronic Computers, Vol. EC-8, No. 3, pp. 330-334, 1959.

[2] J. S. Walter, "A Unified Algorithm for Elementary Functions", AFIPS Conf. Proc., Vol. 38, pp. 379-385, 1971.

[3] Y. H. Hu, "The quantization effects of the CORDIC algorithm", IEEE Transactions on Signal Processing, Vol. 40 I. 4, pp. 834-844, 1992.

[4] R. Andraka, "A Survey of CORDIC Algorithms for FPGA Based Computers", Proc. ACM/SIGDA Sixth Int. Symposium on Field Programmable Gate Arrays, pp. 191-200, 1998.

[5] Y. H. Hu, H. M. Chern, "A novel implementation of CORDIC algorithm using Backward Angle Recoding (BAR)", IEEE Transactions on Computers Vol. 45. I. 12, pp. 1370-1378, 1996.