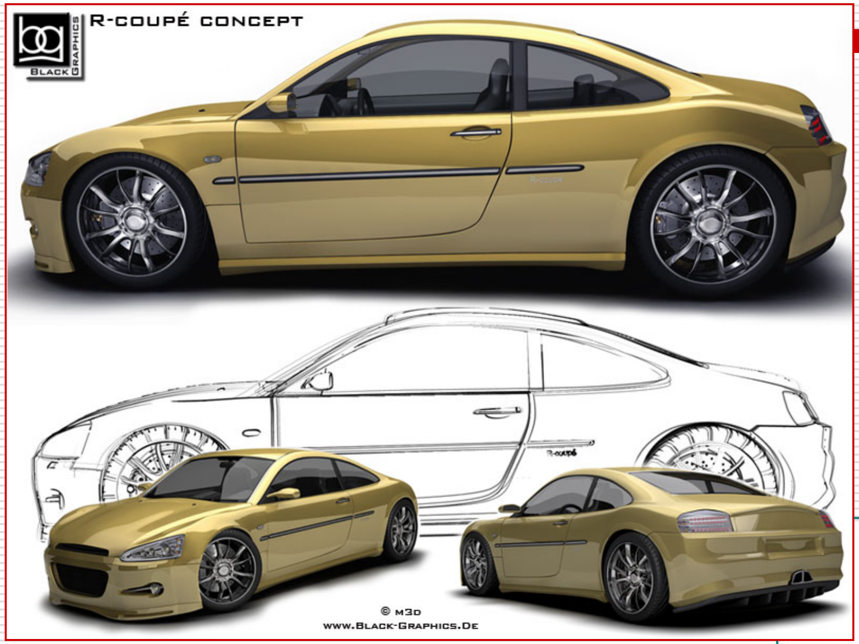
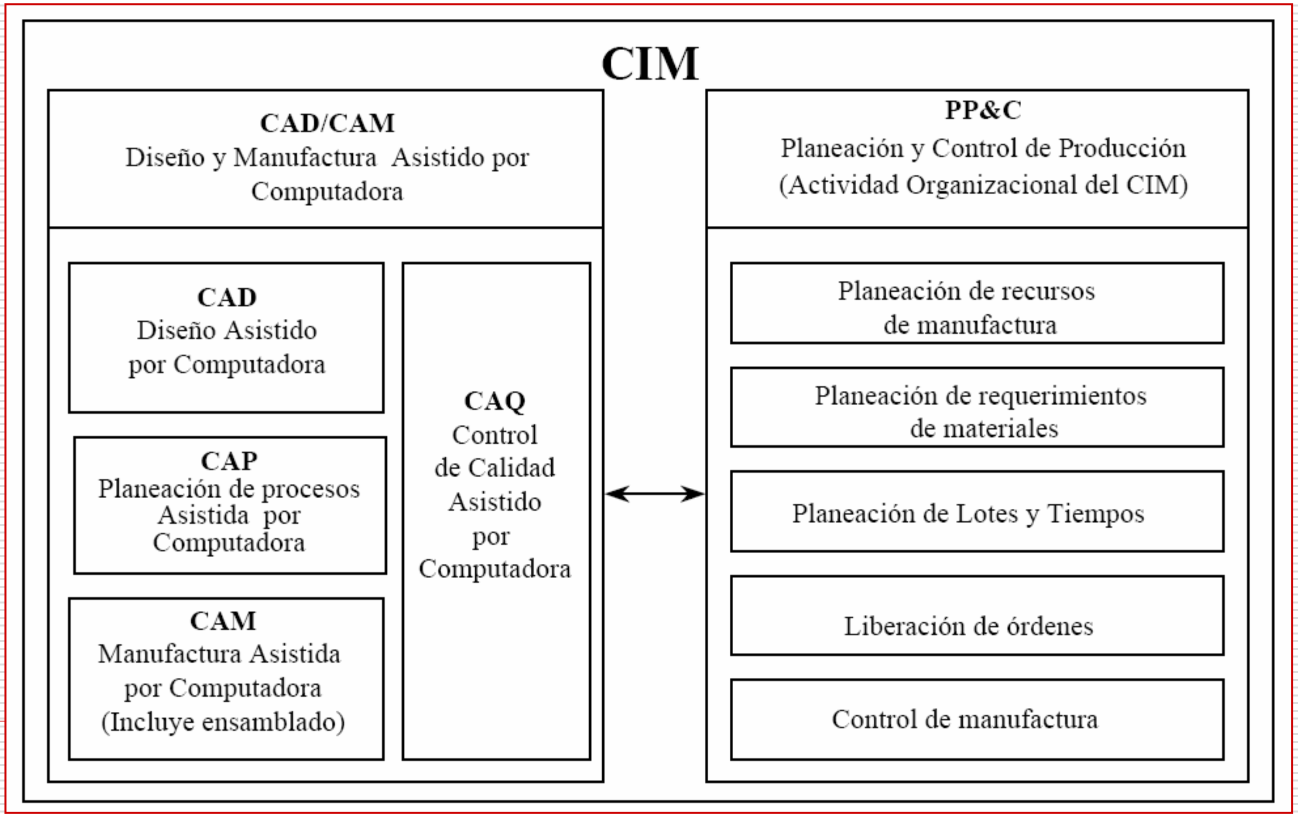
DEPARTAMENTO DE TECNOLOGÍA INFORMÁTICA Y COMPUTACIÓN

TEMA 2: ARQUITECTURAS PARA CAD/CAM



2.1. Introducción

- CAD/CAM : Diseño Asistido por Computador / Fabricación Asistida por Computador
- Se incluyen dentro de la CIM (*Fabricación Integrada por Computador*), tecnología cooperativa orientada a diseño, fabricación y control de la producción

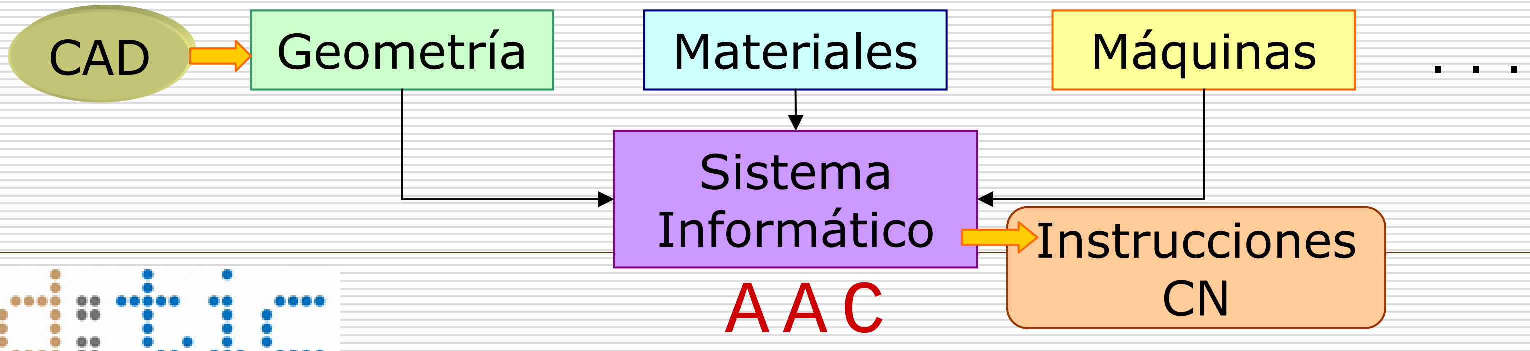


2.1. Introducción

- CAD : Utilización de sistemas informáticos para creación, modificación, análisis y optimización de un diseño
  - Se considera que una aplicación es CAD si incluye una interfaz gráfica y se aplica a alguna tarea de ingeniería
  - Herramientas de todo tipo: modelado, cálculo de propiedades físicas, etc.
  - El CAGD (Diseño Geométrico) es un caso particular del CAD, del que podemos distinguir dos variantes:
    - Modelado de sólidos: diseño de objetos poliédricos; orientado al diseño mecánico y arquitectónico
    - Diseño de curvas y superficies: orientado a la producción industrial (automóvil –Renault-, naval, aeronáutica)

2.1. Introducción

- CAM : Utilización de sistemas informáticos para la planificación, gestión y control de las operaciones de una planta de fabricación
  - Dos tipos principales de interfaz entre el sistema informático y los recursos de producción
    - Interfaz directa. Conexión directa al proceso de producción para monitorizar la actividad (supervisión y control)
    - Interfaz indirecta. El sistema informático constituye una ayuda para la fabricación, pero no ejerce control directo
  - Control numérico: Programación de máquinas herramienta para transformar la materia prima en producto terminado



2.1. Introducción

- CAM
  - Programación de robots. Selección y posicionamiento de piezas y herramientas para las máquinas de control numérico
  - El proceso CAM:

Modelo geométrico
↓
Algoritmos de interfaz
↓
Planificación de procesos
↓
Programación control numérico
↓
Inspección
↓
Ensamblado
↓
Embalaje

2.1. Introducción

- Antecedentes históricos de las arquitecturas para CAD/CAM

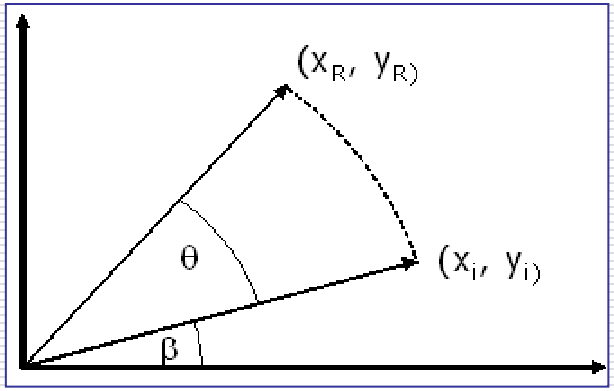
Años 50 y 60	Ordenadores de gran tamaño (100 millones $) Primera pantalla gráfica (MIT) Periféricos para gráficos interactivos (lápiz) Aparición de pantallas de ordenador comerciales
Años 70	Minicomputadores (*Cabinas*) de millón $
Principios 80	Incremento de potencia (32 bits)
Finales 80	Sistemas potentes basados en estaciones UNIX
Principios 90	Estaciones PC
Finales 90- s. XXI	Internet e Intranets Aumenta la potencia y disminuye el coste del hardware Se impone el PC

2.2. El algoritmo CORDIC

- En las aplicaciones CAD/CAM existe ciertas operaciones muy específicas que se repiten con bastante frecuencia (raíces, distancias, rotaciones).
- Estas operaciones son costosas en arquitecturas de propósito general.
- ¿Existen algoritmos o arquitecturas que permitan optimizar el cálculo de tales operaciones? Se busca precisión y eficiencia

2.2. El algoritmo CORDIC

- COordinate Rotation DIgital Computer : algoritmo concebido inicialmente para cálculo de rotaciones y otros problemas de trigonometría en computadores para navegación aérea (1959)
- Supongamos un sistema de coordenadas circular (2D) en el que queremos rotar un punto (x_i, y_i) un ángulo θ



Rotación perfecta:

$$x_R = M_i \cos(\beta + \theta) = x_i \cos\theta - y_i \sin\theta$$

$$y_R = M_i \operatorname{sen}(\beta + \theta) = x_i \operatorname{sen}\theta + y_i \cos\theta$$

$$\begin{bmatrix} x_R \\ y_R \end{bmatrix} = \begin{bmatrix} \cos\theta & -\operatorname{sen}\theta \\ \operatorname{sen}\theta & \cos\theta \end{bmatrix} \begin{bmatrix} xi \\ yi \end{bmatrix} = ROT(\theta) \begin{bmatrix} xi \\ yi \end{bmatrix}$$

2.2. El algoritmo CORDIC

- Podemos utilizar ángulos elementales α_j para descomponer el ángulo de rotación θ, tal que

$$\theta = \sum_{j=0}^{\infty} \alpha_j \Rightarrow ROT(\theta) = \prod_{j=0}^{\infty} ROT(\alpha_j)$$

- Entonces, tenemos que

$$x_R[j+1] = x_R[j] \cos(\alpha_j) - y_R[j] \operatorname{sen}(\alpha_j)$$
$$y_R[j+1] = x_R[j] \operatorname{sen}(\alpha_j) + y_R[j] \cos(\alpha_j)$$

2.2. El algoritmo CORDIC

- ¿Cómo podemos evitar los productos?

  1. Descomponer las rotaciones en:
     - Escalado y rotación-extensión (pseudorotación)

$$x_R[j+1] = \cos(\alpha_j)\ (\ x_R[j] - y_R[j]\tan(\alpha_j)\ )$$
$$y_R[j+1] = \cos(\alpha_j)\ (\ y_R[j] + x_R[j]\tan(\alpha_j)\ )$$

  2. Escoger ángulos elementales:

$$\alpha_j = \tan^{-1}(\sigma_j(2^{-j})) = \sigma_j \tan^{-1}(2^{-j}) \quad , \sigma_j \in \{-1,1\}$$

- Así, aparece recurrencia en rotaciones-extensiones sin productos; SÓLO SUMAS Y DESPLAZAMIENTOS

$$x[j+1] = x[j] - \sigma_j\, 2^{-j}\, y[j]$$
$$y[j+1] = y[j] + \sigma_j\, 2^{-j}\, x[j]$$

2.2. El algoritmo CORDIC

- Las rotaciones-extensiones producen un escalado del módulo M[j]

$$M[j+1] = K[j]\,M[j] = \frac{1}{\cos(\alpha_j)}\,M[j] = (1 + \sigma_j^2\, 2^{-2j})^{1/2}\,M[j] =$$

$$= (1 + 2^{-2j})^{1/2}\,M[j]$$

- El factor de escalado total es:

$$\boxed{K = \prod_{j=0}^{\infty} (1 + 2^{-2j})^{1/2} \approx 1.6468}$$

- Se trata de un factor constante e independiente del ángulo de rotación

2.2. El algoritmo CORDIC

- La recurrencia para descomposición/acumulación del ángulo es:

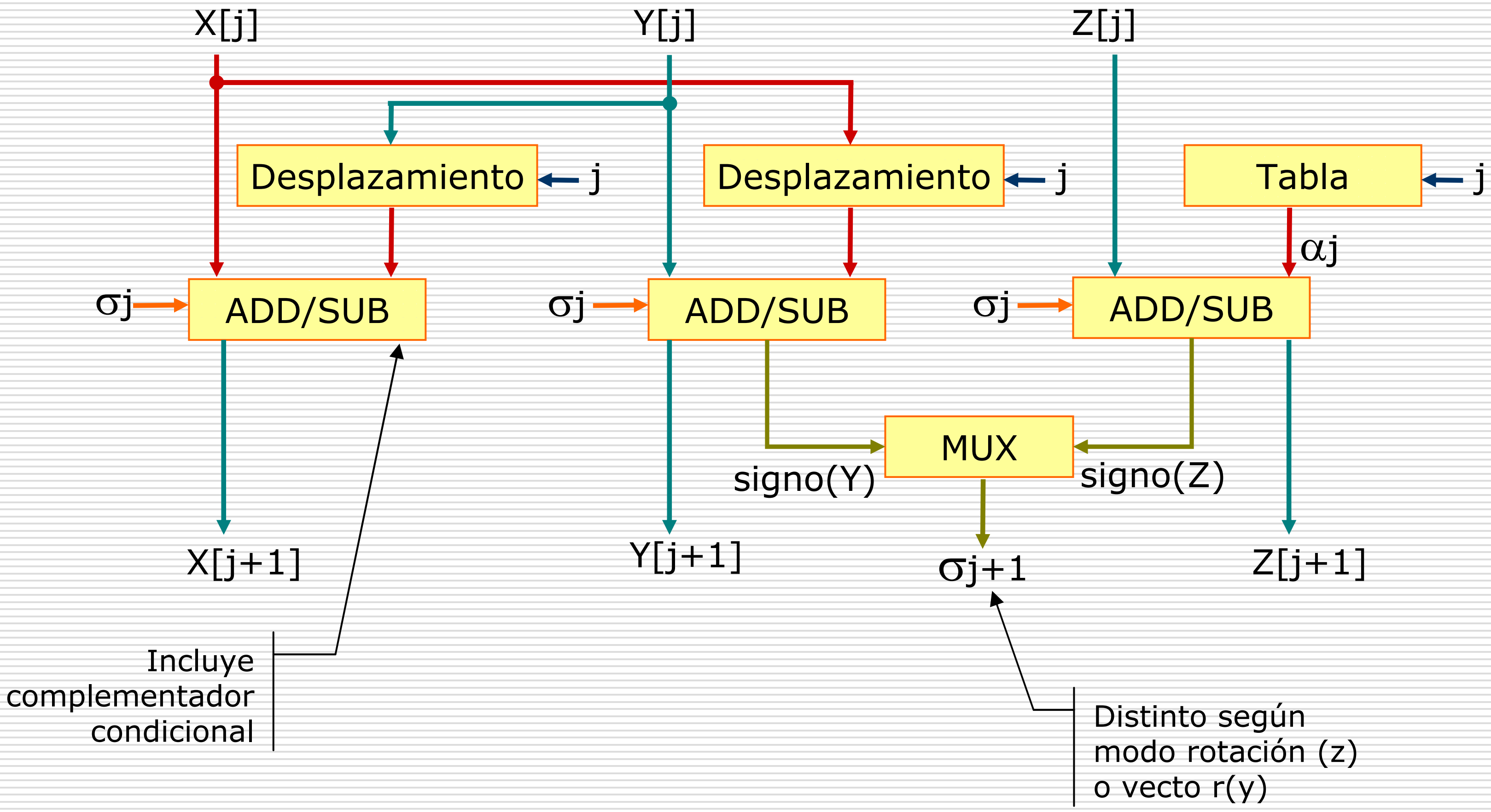
$$z[j+1] = z[j] - \alpha_j = z[j] - \sigma_j \tan^{-1}(2^{-j})$$

- Por tanto, una micro-rotación CORDIC se calcula:

$$x[j+1] = x[j] - \sigma_j 2^{-j} y[j]$$
$$y[j+1] = y[j] + \sigma_j 2^{-j} x[j]$$
$$z[j+1] = z[j] - \sigma_j \tan^{-1}(2^{-j})$$

2.2. El algoritmo CORDIC

- Arquitectura de una iteración



2.2. El algoritmo CORDIC

- El modo *rotación*
  - Se trata de rotar un vector inicial (x_i, y_i) por un ángulo θ
  - Se descompone el ángulo:

$$z[j+1] = z[j] - \alpha_j = z[j] - \sigma_j \tan^{-1}(2^{-j})$$

$$x[0] = x_i,\ \ y[0] = y_i,\ \ z[0] = \theta$$

$$\sigma_j = \begin{cases} 1 & \text{si } z[j] \geq 0 \\ -1 & \text{si } z[j] < 0 \end{cases}$$

  - Realiza micro-rotaciones, y los valores finales son:

$$x_f = K\,(x_i \cos\theta - y_i \sin\theta)$$

$$y_f = K\,(x_i \,\text{sen}\,\theta + y_i \cos\theta)$$

$$z_f = 0$$

2.2. El algoritmo CORDIC

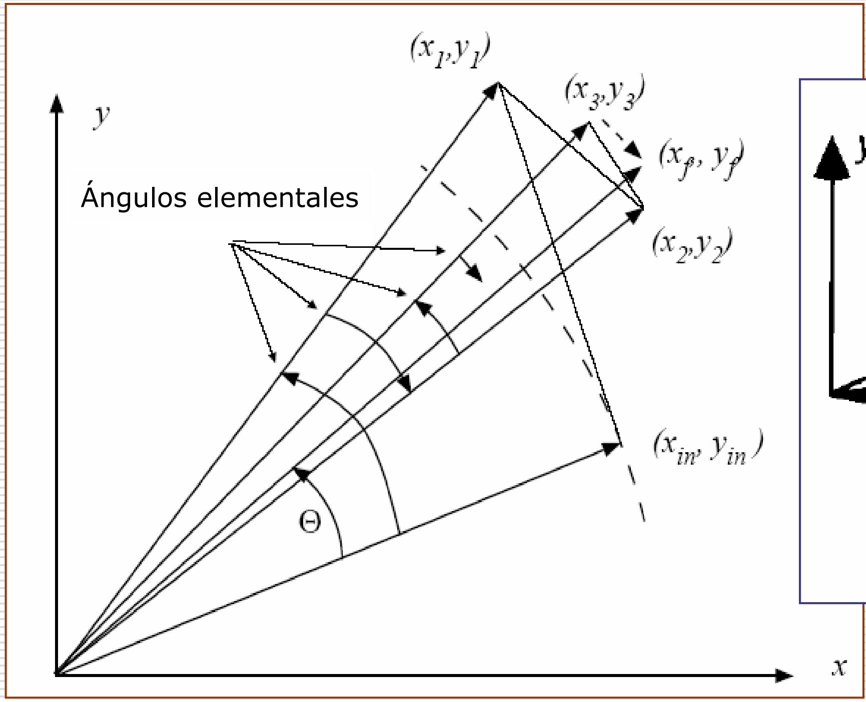
- El modo *rotación*
  - Ejemplo: $(x_i, y_i) = (1, 0.125)$, $\theta = 67º \cong 1.1693$ rad
  - Solución exacta: $(x_f, y_f) = (0.2756, 0.9693)$

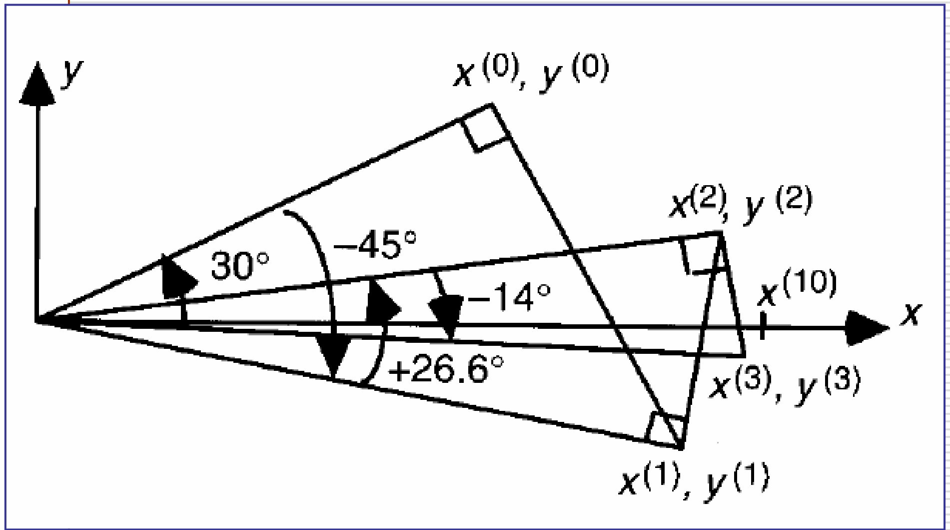
j	z[j]	σj	x[j]	y[j]
0	1.1693	1	1.0	0.125
1	0.3839	1	0.875	1.125
2	-0.0796	-1	0.3125	1.1562
3	0.1653	1	0.7031	1.4843
4	0.0409	1	0.5175	1.5722
5	-0.0214	-1	0.4193	1.6046
6	0.0097	1	0.4694	1.5915

j	z[j]	σj	x[j]	y[j]
7	-0.0058	-1	0.4445	1.5988
8	0.0019	1	0.4570	1.5953
9	-0.0019	-1	0.4508	1.5971
10	0.0000	1	0.4539	1.5962
11	-0.0009	-1	0.4524	1.5967
12	-0.0004	-1	0.4531	1.5965
13			0.4535	1.5963

2.2. El algoritmo CORDIC

- El modo *rotación*
  - Factor de escalado, K = 1.64676
  - $(x_{13}, y_{13})\frac{1}{K} = (0.2753, 0.9693)$, errores $< 2^{-12}$





2.2. El algoritmo CORDIC

- El modo *rotación*
  - Cálculo de funciones trigonométricas

$$x[0] = 1/K,\ \ y[0] = 0,\ \ \ z[0] = \theta$$

$$x_f = \cos\theta$$
$$y_f = \mathrm{sen}\ \theta$$
$$z_f = 0$$

  - En general, si hacemos

$$x[0] = a/K,\ \ y[0] = b/K,\ \ \ z[0] = \theta$$

$$x_f = a \cos\theta - b\ \mathrm{sen}\ \theta$$
$$y_f = a\ \mathrm{sen}\ \theta - b \cos\theta$$
$$z_f = 0$$

2.2. El algoritmo CORDIC

- El modo *vector (vectoring)*
  - Se trata de rotar un vector inicial (x_i, y_i) hasta que $y = 0$

$$\sigma_j = \begin{cases} 1 & \text{si } y[j] < 0 \\ -1 & \text{si } y[j] \geq 0 \end{cases}$$

  - Se acumula el ángulo rotado en z

$$x[0] = x_i, \quad y[0] = y_i, \quad z[0] = z_i$$

$$x_f = K(x_i^2 + y_i^2)^{1/2}$$
$$y_f = 0$$
$$z_f = z_i + \tan^{-1}(y_i/x_i)$$

2.2. El algoritmo CORDIC

- El modo *vector*
  - Ejemplo: $(x_i, y_i) = (0.75, 0.43)$
  - Forzamos y = 0 en 12 micro-rotaciones
  - Solución exacta: $x_R = (x_i^2 + y_i^2)^{1/2} = 0.8645$
  - $z_R = \tan^{-1}\left(\frac{0.43}{0.75}\right) = 0.5205$

j	y[j]	σj	x[j]	z[j]
0	0.43	-1	0.75	0.0
1	-0.32	1	1.18	0.7853
2	0.27	-1	1.34	0.3217
3	-0.065	1	1.4075	0.5667
4	0.1109	-1	1.4156	0.4423
5	0.0224	-1	1.4225	0.5047
6	-0.0219	1	1.4232	0.5360

j	y[j]	σj	x[j]	z[j]
7	0.0002	-1	1.4236	0.5204
8	-0.0108	1	1.4236	0.5282
9	-0.0053	1	1.4236	0.5243
10	-0.0025	1	1.4236	0.5223
11	-0.0011	1	1.4236	0.5213
12	-0.0004	1	1.4236	0.5208
13			1.4236	0.5206

2.2. El algoritmo CORDIC

- El modo *vector*
  - Ejemplo: $(x_i, y_i) = (0.75, 0.43)$
  - Ángulo acumulado z[13] = 0.5206
  - $K = 1.64676 \Rightarrow X[13]/K = 0.864$, errores $< 2^{-12}$

j	y[j]	σj	x[j]	z[j]
0	0.43	-1	0.75	0.0
1	-0.32	1	1.18	0.7853
2	0.27	-1	1.34	0.3217
3	-0.065	1	1.4075	0.5667
4	0.1109	-1	1.4156	0.4423
5	0.0224	-1	1.4225	0.5047
6	-0.0219	1	1.4232	0.5360

j	y[j]	σj	x[j]	z[j]
7	0.0002	-1	1.4236	0.5204
8	-0.0108	1	1.4236	0.5282
9	-0.0053	1	1.4236	0.5243
10	-0.0025	1	1.4236	0.5223
11	-0.0011	1	1.4236	0.5213
12	-0.0004	1	1.4236	0.5208
13			1.4236	0.5206

2.2. El algoritmo CORDIC

- Estudio de convergencia, rango y precisión
  - La convergencia está garantizada para ángulos en el rango [-99.7º, 99.7º], que resulta de la suma

$$\theta_{max} = \sum_{j=0}^{\infty} \tan^{-1}\left(2^{-j}\right)$$

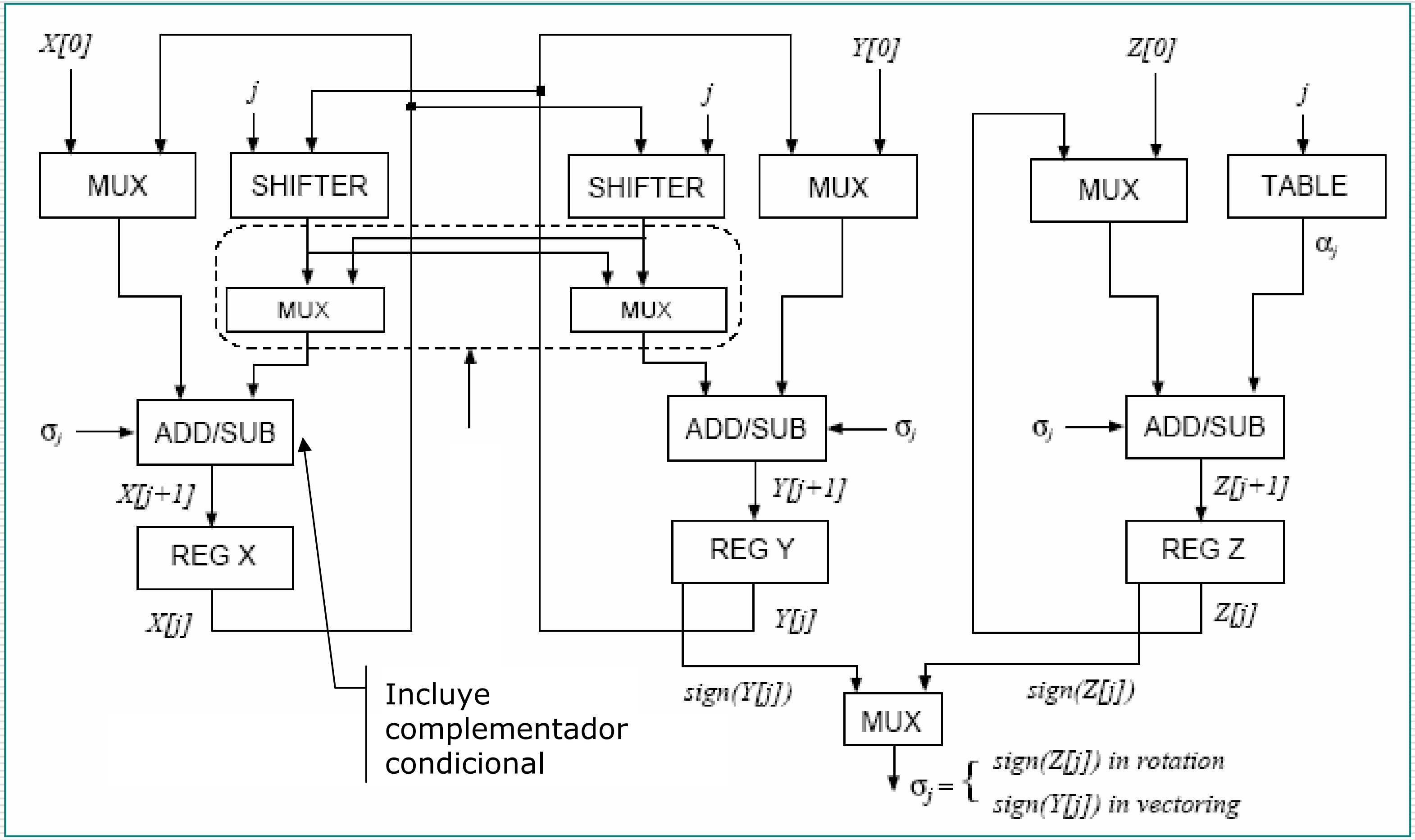
  - Se tiene convergencia al cumplirse que

$$\tan^{-1}(2^{-i}) \leq \sum_{j=i+1}^{\infty} \tan^{-1}(2^{-j})$$

  - Para *k* bits de precisión, se necesitan *k* iteraciones.
  - Paa grandes i, $\tan^{-1}(2^{-i}) \cong 2^{-i}$

2.2. El algoritmo CORDIC

- Arquitectura palabra-serie

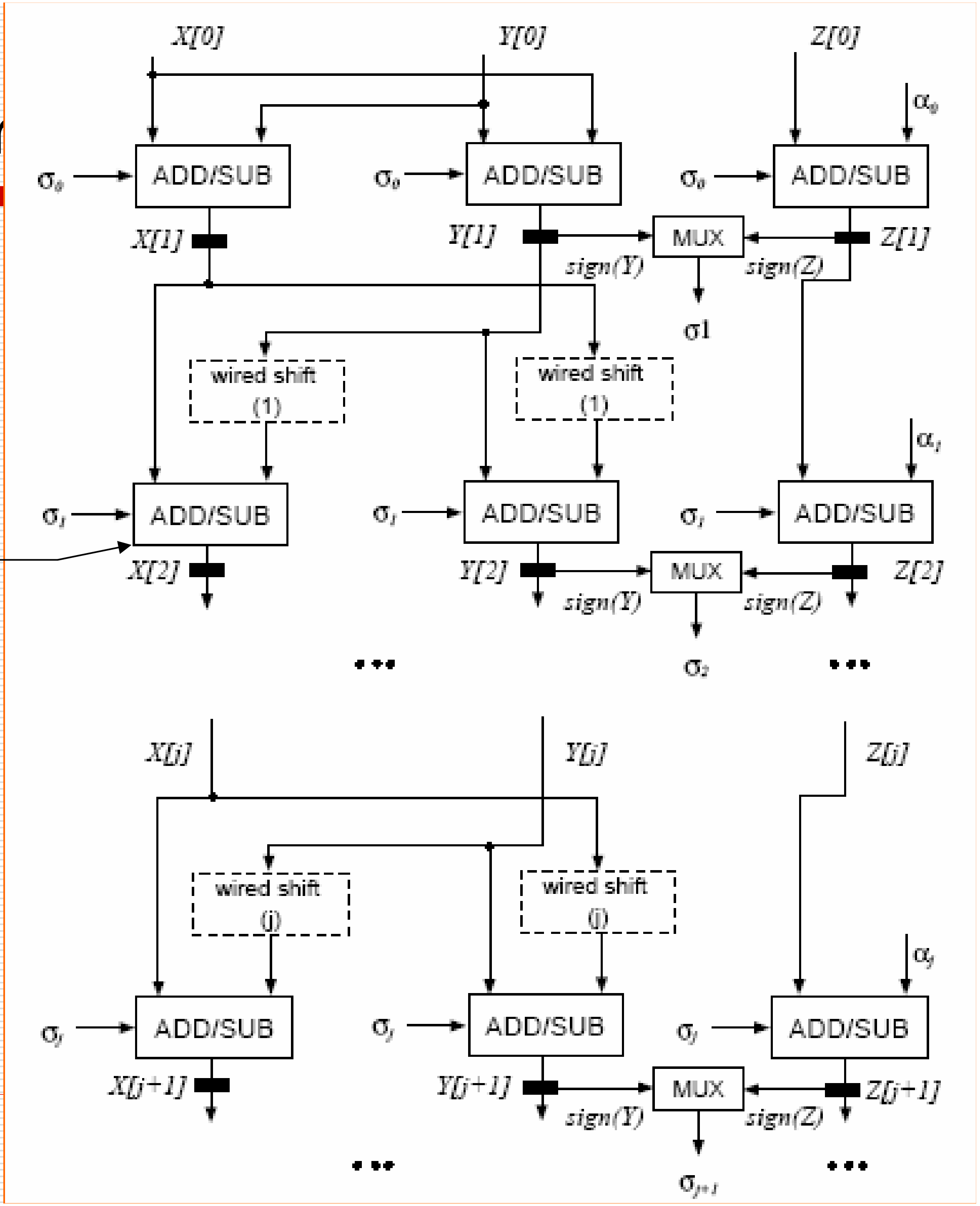


2.2. El algoritm

- Arquitectura

pipeline

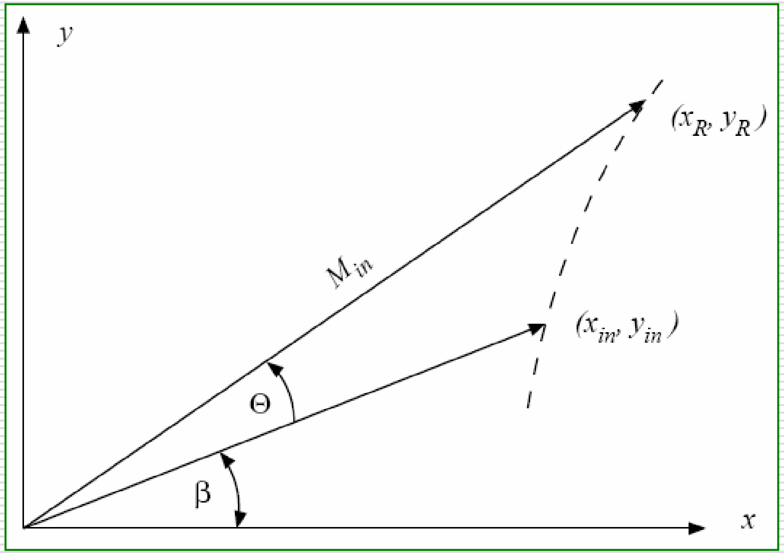
Incluye
complementador
condicional



2.2. El algoritmo CORDIC

- Extensión a coordenadas hiperbólicas

$$\begin{bmatrix} x_R \\ y_R \end{bmatrix} = \begin{bmatrix} \cosh\theta & -\text{senh}\,\theta \\ \text{senh}\,\theta & \cosh\theta \end{bmatrix} \begin{bmatrix} xi \\ yi \end{bmatrix}$$



- Las micro-rotaciones hiperbólicas serían:

$$x[j+1] = x[j] + \sigma_j\, 2^{-j}\, y[j]$$
$$y[j+1] = y[j] + \sigma_j\, 2^{-j}\, x[j]$$
$$z[j+1] = z[j] - \sigma_j \tanh^{-1}(2^{-j})$$

- El factor de escalado en la iteración j resulta:

$$K_h[j] = (1 - 2^{-2j})^{1/2}$$

- Ya que $\tanh^{-1}(2^0) = \infty$ (y K_h [0]=0), se ha de comenzar desde la iteración j = 1

2.2. El algoritmo CORDIC

- Extensión a coordenadas hiperbólicas
  - Problema: no hay convergencia con la secuencia de ángulos $\tanh^{-1}(2^{-j})$, ya que:

$$\tanh^{-1}(2^{-i}) > \sum_{j=i+1}^{\infty} \tanh^{-1}(2^{-j})$$

  - Solución sencilla: repetir ciertas iteraciones:

$$\sum_{j=i+1}^{\infty} \tanh^{-1}(2^{-j}) < \tanh^{-1}(2^{-i}) < \sum_{j=i+1}^{\infty} \tanh^{-1}(2^{-j}) + \tanh^{-1}(2^{-(3j+1)})$$

  - Existe convergencia al repetir las iteraciones 4, 13, 40, ..., 3k + 1
  - Con estas iteraciones, $K_h \cong 0.82816$, $\theta_{max} = 1.11817$ rad

2.2. El algoritmo CORDIC

- Extensión a coordenadas hiperbólicas
  - Valores finales:
    - Modo rotación

$$x_f = K_h\,(x_i \cosh\theta + y_i \sinh\theta)$$

$$y_f = K_h\,(x_i \,\mathrm{senh}\,\theta + y_i \cosh\theta)$$

$$z_f = 0$$

    - Modo vector

$$x_f = K_h(x_i^2 - y_i^2)^{1/2}$$

$$y_f = 0$$

$$z_f = z_i + \tanh^{-1}(y_i/x_i)$$

2.2. El algoritmo CORDIC

- Extensión a coordenadas lineales
  - Las micro-rotaciones serían (factor de escalado = 1):

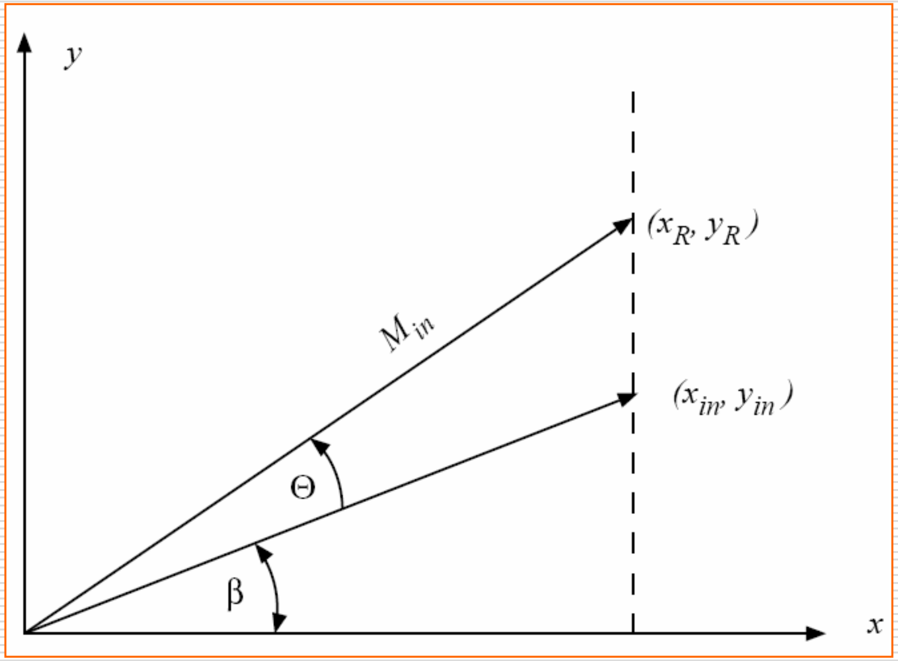
$$x[j+1] = x[j]$$
$$y[j+1] = y[j] + \sigma_j\, 2^{-j}\, x[j]$$
$$z[j+1] = z[j] - \sigma_j\, 2^{-j}$$

- Modo rotación

$$x_f = x_i$$
$$y_f = y_i + x_i z_i$$
$$z_f = 0$$

- Modo vector

$$x_f = x_i$$
$$y_f = 0$$
$$z_f = z_i + y_i/x_i$$



2.2. El algoritmo CORDIC

- Descripción unificada del algoritmo:
  - Coordenas circulares: $m = 1$
  - Coordenadas hiperbólicas: $m = -1$
  - Coordenadas lineales: $m = 0$
  - Modo rotación: $\sigma_j = \text{signo}(z[j]) \in \{-1, 1\}$
  - Modo vector: $\sigma_j = - \text{signo}(y[j]) \in \{-1, 1\}$

$$x[j+1] = x[j] - m\,\sigma_j\,2^{-j}\,y[j]$$
$$y[j+1] = y[j] + \sigma_j\,2^{-j}\,x[j]$$

$$z[j+1] = \begin{cases} z[j] - \sigma_j \tan^{-1}(2^{-j}) & \text{si } m = 1 \\ z[j] - \sigma_j \tanh^{-1}(2^{-j}) & \text{si } m = -1 \\ z[j] - \sigma_j\, 2^{-j} & \text{si } m = 0 \end{cases}$$

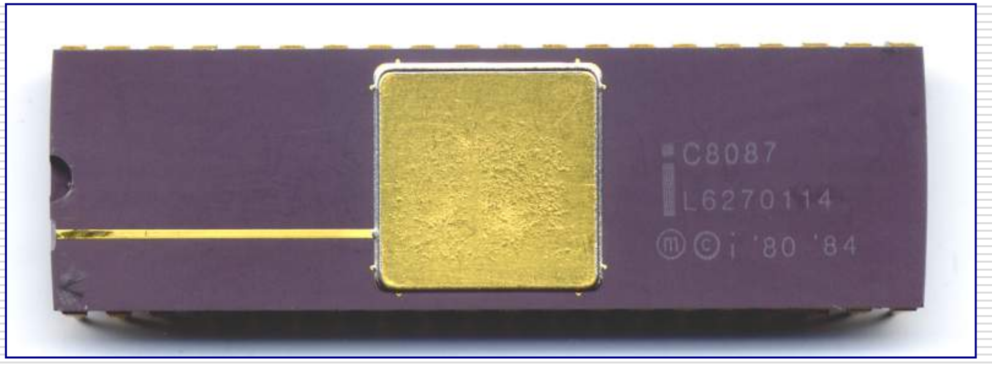
2.2. El algoritmo CORDIC

- Otras funciones que podemos calcular:

m	Modo	Valores iniclaes			x_R	y_R	z_R
		x_i	y_i	z_i			
1	rotación	1	0	θ	cos θ	sen θ	
-1	rotación	1	0	θ	cosh θ	senh θ	
-1	rotación	a	a	θ	$a\ e^{\theta}$	$a\ e^{\theta}$	
1	vector	1	a	$\pi/2$	$(a^2+1)^{1/2}$		$\cot^{-1}(a)$
-1	vector	a	1	0	$(a^2-1)^{1/2}$		$\coth^{-1}(a)$
-1	vector	a+1	a-1	0	$2\ a^{1/2}$		$0.5\ \ln(a)$
-1	vector	a+1/4	a-1/4	0	$a^{1/2}$		$\ln(a/4)$
-1	vector	a+b	a-b	0	$2\ (ab)^{1/2}$		$0.5\ \ln(a/b)$

2.2. El algoritmo CORDIC

- Utilización del CORDIC:
  - Calculadoras: HP 35 (1972)
  - Coprocesadores: Intel 8087 (1980)



  - No se ha utilizado de forma masiva…
  - ¿Ventajas?
  - ¿Inconvenientes?