

CORDIC Background (2A)

-
-

Copyright (c) 2010, 2011 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

CORDIC Background

1.A survey of CORDIC algorithms for FPGAs, Ray Andraka,
www.andraka.com/cordic.htm

Vector Rotation (1)

$$x' = x \cos \phi - y \sin \phi$$

$$y' = y \cos \phi + x \sin \phi$$

$$x' = \cos \phi \cdot [x - y \tan \phi]$$

$$y' = \cos \phi \cdot [y + x \tan \phi]$$

$$x_{i+1} = K_i \cdot [x_i - y_i \cdot d_i \cdot 2^{-i}]$$

$$y_{i+1} = K_i \cdot [y_i + x_i \cdot d_i \cdot 2^{-i}]$$

$$K_i = \cos \phi_i = \cos(\tan^{-1}(2^{-i}))$$

$$= \frac{1}{\sqrt{1 + 2^{-2i}}}$$

$$d_i = \pm 1$$

Restrict rotation angle $\Rightarrow \tan \phi = \pm 2^{-i}$

Multiplication \Rightarrow simple shift

$$y \cdot \tan \phi$$

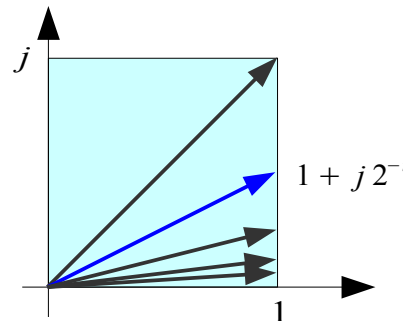
$$y \cdot 2^{-i}$$

$$x \cdot \tan \phi$$

$$x \cdot 2^{-i}$$

regardless of direction $\Rightarrow \cos(\phi) = \cos(-\phi)$

Allowed Rotation Angles



$$\tan \phi \Rightarrow 2^{-i}$$

$$\cos \phi \Rightarrow \frac{1}{\sqrt{1 + 2^{-2i}}}$$

$$K_i \leq 1$$

$$(K_i \leftarrow \cos \phi)$$

Vector Rotation (2)

$$x_{i+1} = K_i \cdot [x_i - y_i \cdot d_i \cdot 2^{-i}]$$

$$y_{i+1} = K_i \cdot [y_i + x_i \cdot d_i \cdot 2^{-i}]$$

$$K_i = \cos \phi_i = \cos(\tan^{-1}(2^{-i}))$$

$$= \frac{1}{\sqrt{1 + 2^{-2i}}} \quad K_i \leq 1$$

$$d_i = \pm 1$$

$$x_{i+1}^2 = K_i^2 \cdot [x_i^2 + y_i^2 \cdot 2^{-2i} - 2x_i y_i d_i \cdot 2^{-i}]$$

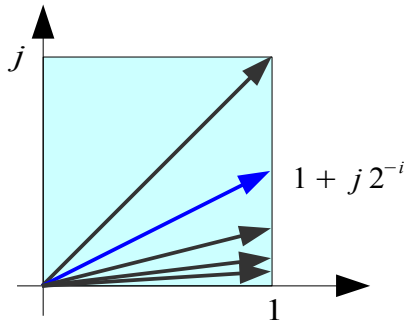
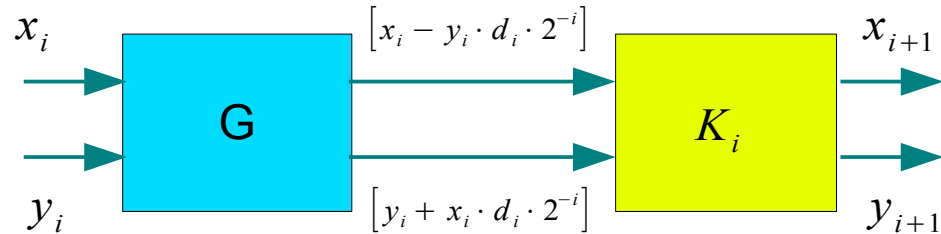
$$y_{i+1}^2 = K_i^2 \cdot [y_i^2 + x_i^2 \cdot 2^{-2i} + 2x_i y_i d_i \cdot 2^{-i}]$$

$$x_{i+1}^2 + y_{i+1}^2 = K_i^2 \cdot (1 + 2^{-2i}) \cdot (x_i^2 + y_i^2)$$

$$G \cdot K_i = 1$$

$$K_i \leq 1$$

$$G > 1$$



$$\tan \phi \rightarrow 2^{-i}$$

$$\cos \phi \rightarrow \frac{1}{\sqrt{1 + 2^{-2i}}}$$

CORDIC Gain : *growing in magnitude*

$$A_n = \prod_{i=1}^n \frac{1}{K_i} = \prod_{i=1}^n \sqrt{1 + 2^{-2i}} \rightarrow 1.647$$

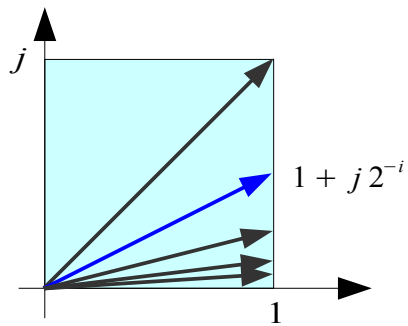
Vector Rotation (3)

$$x_{i+1} = K_i \cdot [x_i - y_i \cdot d_i \cdot 2^{-i}]$$

$$y_{i+1} = K_i \cdot [y_i + x_i \cdot d_i \cdot 2^{-i}]$$

$$K_i = 1 / \sqrt{1 + 2^{-2i}} \quad \leftarrow \cos(\phi_i)$$

$$d_i = \pm 1$$



$$\tan \phi \rightarrow 2^{-i}$$

$$\cos \phi \rightarrow \frac{1}{\sqrt{1 + 2^{-2i}}}$$

Without Scale Constants K_i

$$x_{i+1} = [x_i - y_i \cdot d_i \cdot 2^{-i}]$$

$$y_{i+1} = [y_i + x_i \cdot d_i \cdot 2^{-i}]$$

$$d_i = \pm 1$$

CORDIC Gain : *growing in magnitude*

$$A_n = \prod_{i=1}^n \frac{1}{K_i} = \prod_{i=1}^n \sqrt{1 + 2^{-2i}} \rightarrow 1.647$$

$$1 / K_i = \sqrt{1 + 2^{-2i}}$$

For correction

Multiplying K_i 's as a processing gain

$$\prod_{i=1}^n K_i = \prod_{i=1}^n \frac{1}{\sqrt{1 + 2^{-2i}}} \rightarrow 0.6073$$

Angle Accumulator

Rotation Mode

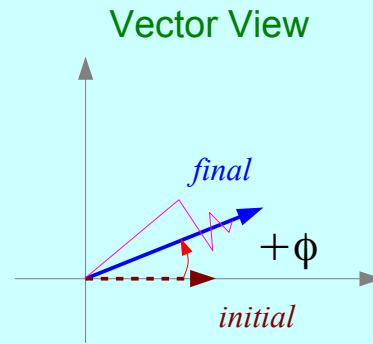
$$z_0 \leftarrow \phi \quad (\text{desired angle})$$

$$z_n \rightarrow 0$$

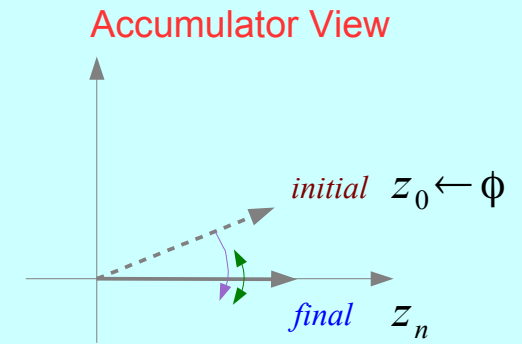
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad \text{if } z_i < 0$$

$$d_i = +1 \quad \text{otherwise}$$



Minimize the residual angle



Subtract angles at each step

Vectoring Mode

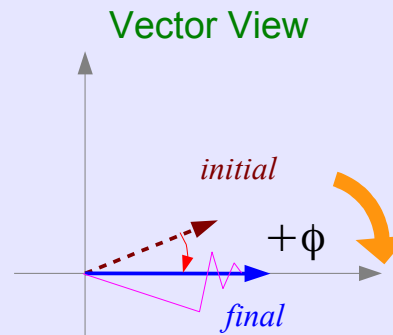
$$z_0 \leftarrow 0$$

$$z_n \rightarrow z_0 + \tan^{-1}(y_0/x_0)$$

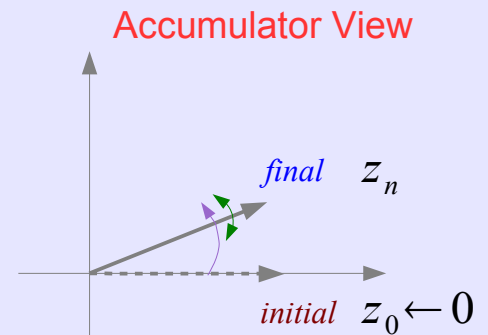
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < 0$$

$$d_i = -1 \quad \text{otherwise}$$



Minimize the residual y component



Add angles at each step

Rotation Mode

Rotation Mode

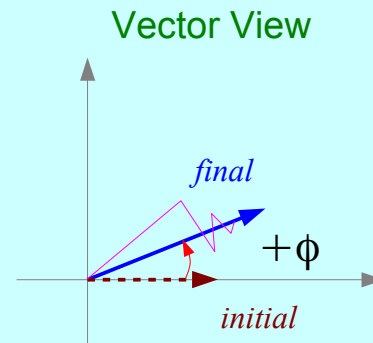
$$z_0 \leftarrow \phi \quad (\text{desired angle})$$

$$z_n \rightarrow 0$$

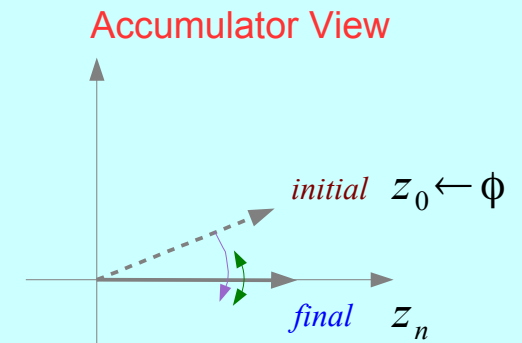
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad \text{if } z_i < 0$$

$$d_i = +1 \quad \text{otherwise}$$



Minimize the residual angle



Subtract angles at each step

$$x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i}$$

$$y_{i+1} = y_i + x_i \cdot d_i \cdot 2^{-i}$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad \text{if } z_i < 0$$

$$d_i = +1 \quad \text{otherwise}$$

$$x_n = A_n [x_0 \cos z_0 - y_0 \sin z_0]$$

$$y_n = A_n [y_0 \cos z_0 + x_0 \sin z_0]$$

$$z_n = 0$$

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

Vectoring Mode

Vectoring Mode

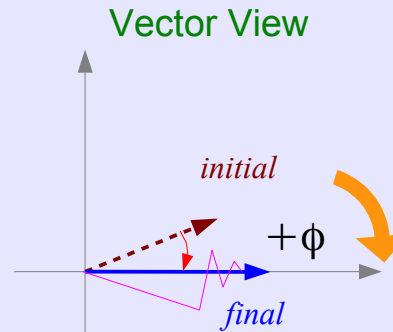
$$z_0 \leftarrow 0$$

$$z_n \rightarrow z_0 + \tan^{-1}(y_0/x_0)$$

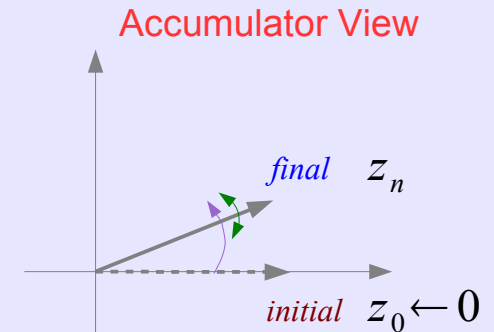
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < 0$$

$$d_i = -1 \quad \text{otherwise}$$



Minimize the residual y component



Add angles at each step

$$x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i}$$

$$y_{i+1} = y_i + x_i \cdot d_i \cdot 2^{-i}$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < 0$$

$$d_i = -1 \quad \text{otherwise}$$

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

$$y_n = 0$$

$$z_n = z_0 + \tan^{-1}(y_0/x_0)$$

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

Angle Accumulator – Rotation Mode

Rotation Mode

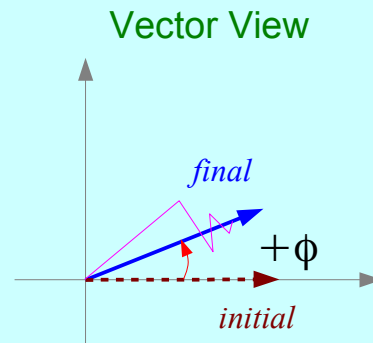
$$z_0 \leftarrow \phi \quad (\text{input})$$

$$z_n \rightarrow 0$$

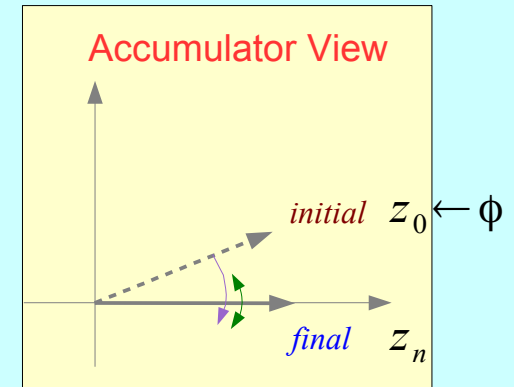
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad \text{if } z_i < 0$$

$$d_i = +1 \quad \text{otherwise}$$



Minimize the residual angle



Subtract angles at each step

$$z_i < 0$$

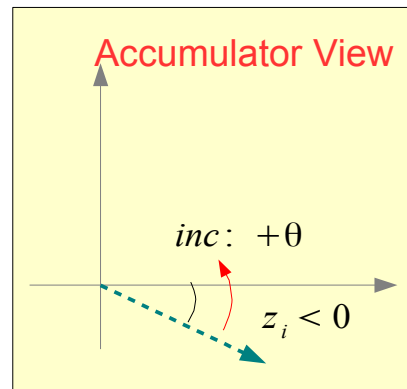
Increase Angle $d_i = -1$

$$z_{i+1} = z_i + \tan^{-1}(2^{-i})$$

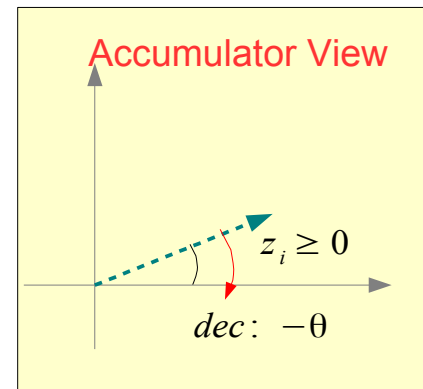
$$z_i \geq 0$$

Decrease Angle $d_i = +1$

$$z_{i+1} = z_i - \tan^{-1}(2^{-i})$$



$z_i < 0$
Increase Angle $+θ$



$z_i \geq 0$
Decreases Angle $-θ$

Angle Accumulator – Vectoring Mode

Vectoring Mode

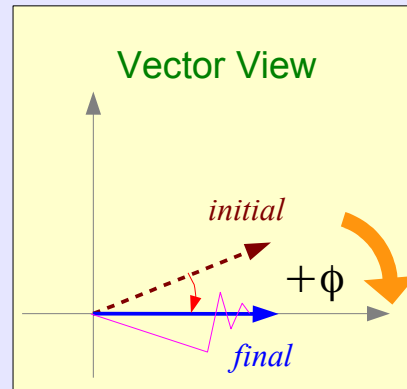
$$z_0 \leftarrow 0$$

$$z_n \rightarrow z_0 + \tan^{-1}(y_0/x_0)$$

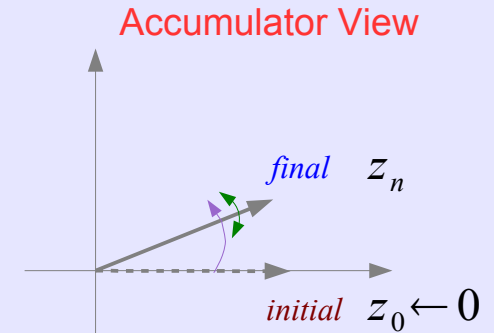
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < 0$$

$$d_i = -1 \quad \text{otherwise}$$



Minimize the residual y component



Add angles at each step

$$y_i < 0$$

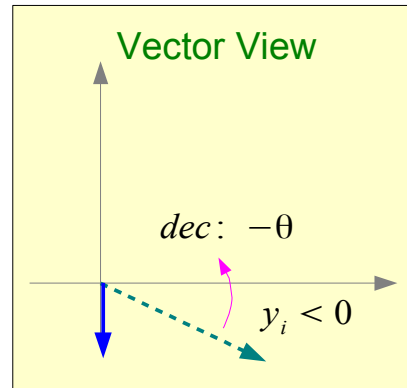
Decrease Angle $d_i = +1$

$$z_{i+1} = z_i - \tan^{-1}(2^{-i})$$

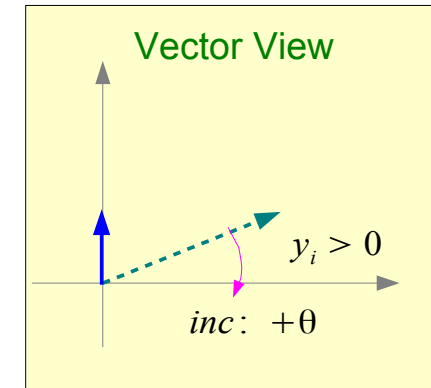
$$y_i > 0$$

Increase Angle $d_i = -1$

$$z_{i+1} = z_i + \tan^{-1}(2^{-i})$$



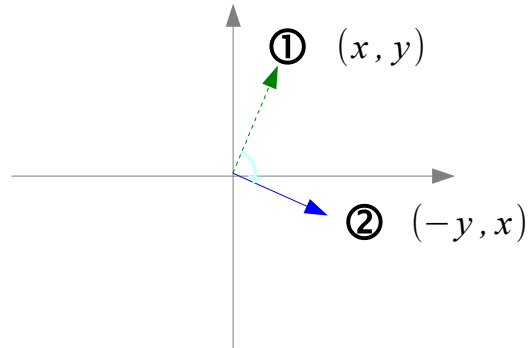
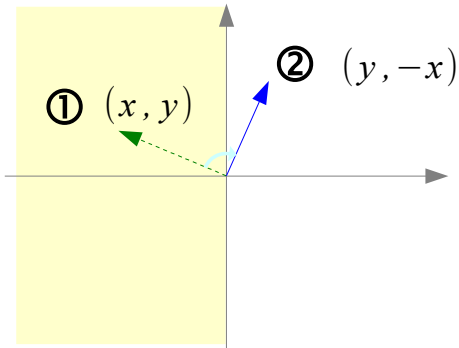
$y_i < 0$
Decrease Angle $-\theta$



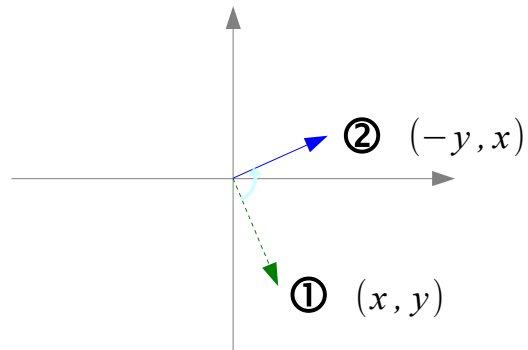
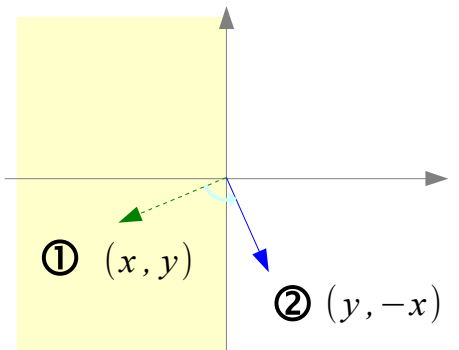
$y_i > 0$
Increases Angle $+\theta$

Initial Rotation $\pm\pi/2$

Positive Phase ($y > 0$) \rightarrow Rotate by -90 degrees



Negative Phase ($y < 0$) \rightarrow Rotate by $+90$ degrees



Resulting Phase \rightarrow $[-90, +90]$

$$\begin{aligned}x' &= -d \cdot y \\y' &= +d \cdot x \\z' &= z + d \cdot \frac{\pi}{2}\end{aligned}$$

$$d = +1 \quad \text{if } y < 0$$

$$d = -1 \quad \text{otherwise}$$

No magnitude change

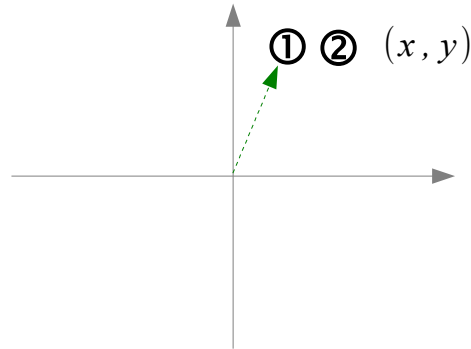
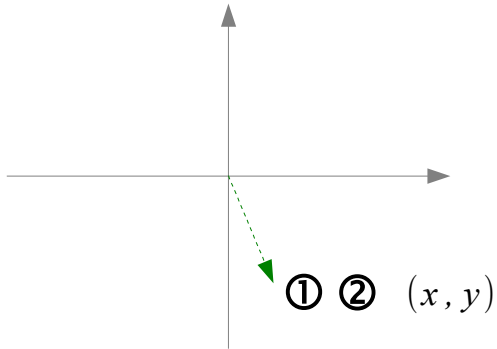
$$x' \leftarrow y$$

$$y' \leftarrow x$$

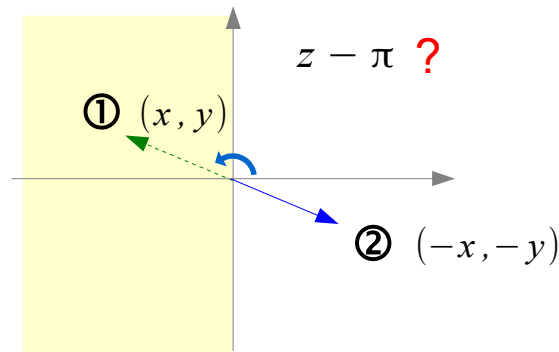
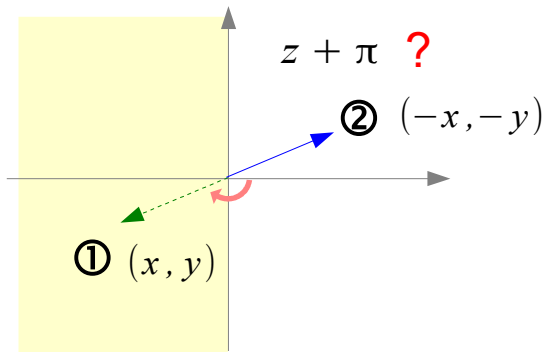
Consistent

Initial Rotation $0, +\pi$

Positive x ($x > 0$) \rightarrow Rotate by 0 degrees



Negative x ($x < 0$) \rightarrow Rotate by $+180$ degrees



Resulting Phase \rightarrow $[-90, +90]$

$$\begin{aligned} x' &= +d \cdot x \\ y' &= +d \cdot y \\ z' &= z \quad \text{if } d = 1 \\ z' &= \pi - z \quad \text{if } d = -1 \end{aligned}$$

$$\begin{aligned} d &= -1 \quad \text{if } x < 0 \\ d &= +1 \quad \text{otherwise} \end{aligned}$$

No magnitude change

$$x' \leftarrow y$$

$$y' \leftarrow x$$

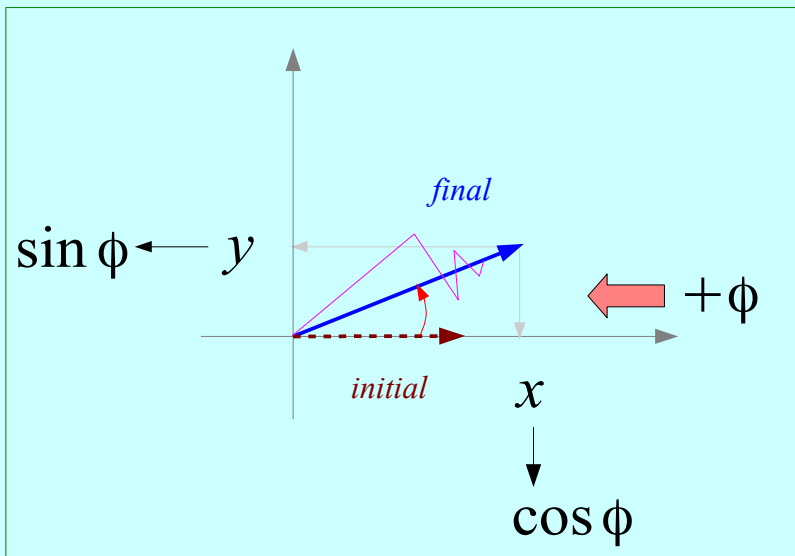
Convenient wiring in
FPGA

Application Modes (1)

Rotation Mode

Input angle is given

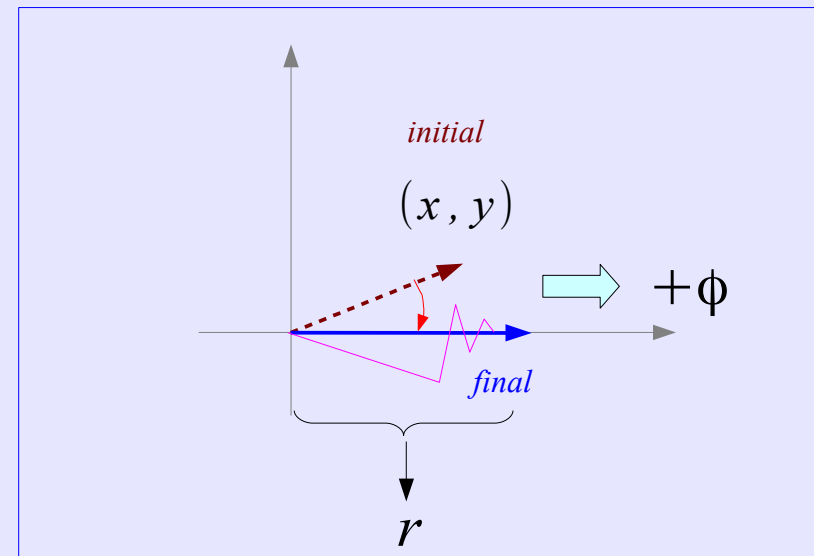
- **sin** and **cos**
- $(r, \theta) \rightarrow (x, y)$
- General vector rotation



Vectoring Mode

Finding the resulting angle

- **\tan^{-1}**
- Vector Magnitude
- $(x, y) \rightarrow (r, \theta)$



Application Modes (2)

- \sin^{-1} and \cos^{-1}
- Inverse CORDIC functions
- Extension to Linear functions
- Extension to Hyperbolic functions

A. Sine and Cosine

Rotation Mode

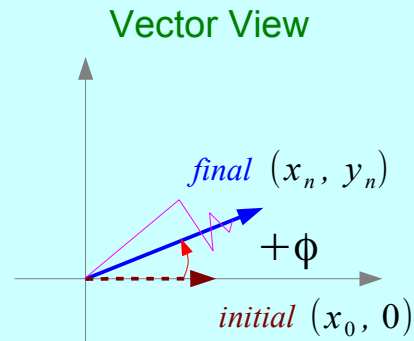
$$z_0 \leftarrow \phi \quad (\text{desired angle})$$

$$z_n \rightarrow 0$$

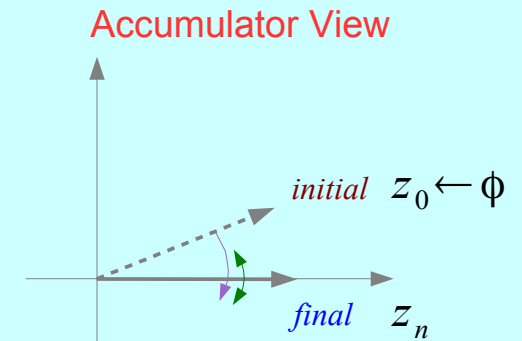
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad \text{if } z_i < 0$$

$$d_i = +1 \quad \text{otherwise}$$



Minimize the residual angle



Subtract angles at each step

Finding Sine and Cosine

$$(x_0, 0) \rightarrow (x_n, y_n)$$

$$x_n = A_n \cdot x_0 \cos z_0$$

$$y_n = A_n \cdot x_0 \sin z_0$$

CORDIC Gain : *growing in magnitude*

$$A_n = \prod_{i=1}^n \frac{1}{K_i} = \prod_{i=1}^n \sqrt{1 + 2^{-2i}} \rightarrow 1.647$$

Unscaled Sine and Cosine

$$x_0 \leftarrow \frac{1}{A_n} = 0.6073$$

$$x_n = \cos z_0$$

$$y_n = \sin z_0$$

Modulated Sine and Cosine

$$x_0 \leftarrow \left\{ \prod_{i=1}^n K_i \right\} \cdot x_0 = 0.6073 \cdot x_0$$

$$x_n = x_0 \cdot \cos z_0$$

$$y_n = x_0 \cdot \sin z_0$$

LUT \rightarrow a pair of MULT

CORDIC \rightarrow rotation operations

Single MULT

B. Polar to Rectangular

Rotation Mode

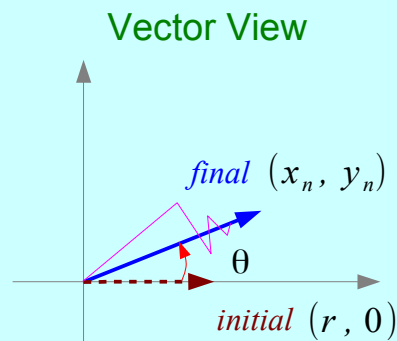
$$z_0 \leftarrow \phi \quad (\text{desired angle})$$

$$z_n \rightarrow 0$$

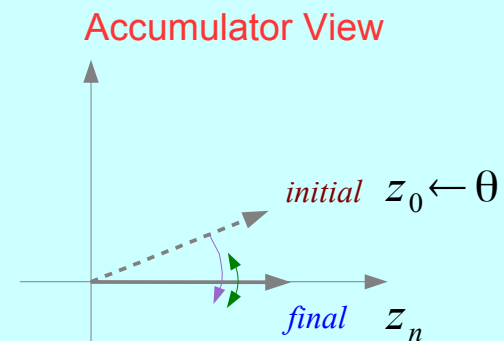
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad \text{if } z_i < 0$$

$$d_i = +1 \quad \text{otherwise}$$



Minimize the residual angle



Subtract angles at each step

Finding Sine and Cosine

$$(x_0, 0) \rightarrow (x_n, y_n)$$

$$x_n = A_n \cdot x_0 \cos z_0$$

$$y_n = A_n \cdot x_0 \sin z_0$$

$$x_0 \leftarrow r, \quad z_0 \leftarrow \theta$$

$$(r, 0) \rightarrow (x_n, y_n)$$

$$x_n = A_n r \cos \theta$$

$$y_n = A_n r \sin \theta$$

$$x_0 \leftarrow r \cdot \frac{1}{A_n}, \quad z_0 \leftarrow \theta$$

$$\left(\frac{r}{A_n}, 0\right) \rightarrow (x_n, y_n)$$

$$x_n = r \cos \theta$$

$$y_n = r \sin \theta$$

CORDIC Gain : *growing in magnitude*

$$A_n = \prod_{i=1}^n \frac{1}{K_i} = \prod_{i=1}^n \sqrt{1 + 2^{-2i}} \rightarrow 1.647$$

C. General Vector Rotation

Rotation Mode

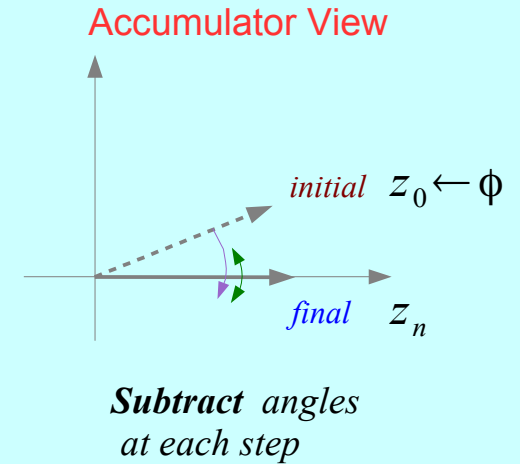
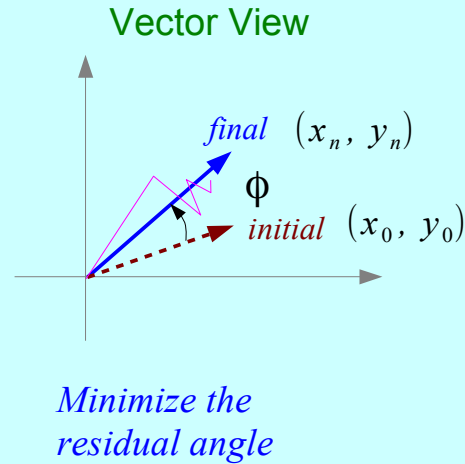
$$z_0 \leftarrow \phi \quad (\text{desired angle})$$

$$z_n \rightarrow 0$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = -1 \quad \text{if } z_i < 0$$

$$d_i = +1 \quad \text{otherwise}$$



Motion Correction and Control System

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = A_n \cdot \begin{bmatrix} \cos z_0 & -\sin z_0 \\ \sin z_0 & \cos z_0 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$$

* n-dim rotation
→ tree architecture

Unscaled Rotation

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = A_n \cdot \begin{bmatrix} \cos z_0 & -\sin z_0 \\ \sin z_0 & \cos z_0 \end{bmatrix} \begin{bmatrix} \frac{x_0}{A_n} \\ \frac{y_0}{A_n} \end{bmatrix} \begin{matrix} \rightarrow \text{A pair of} \\ \rightarrow \text{MULT} \end{matrix} \Rightarrow \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} \cos z_0 & -\sin z_0 \\ \sin z_0 & \cos z_0 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$$

D. Arctangent

Vectoring Mode

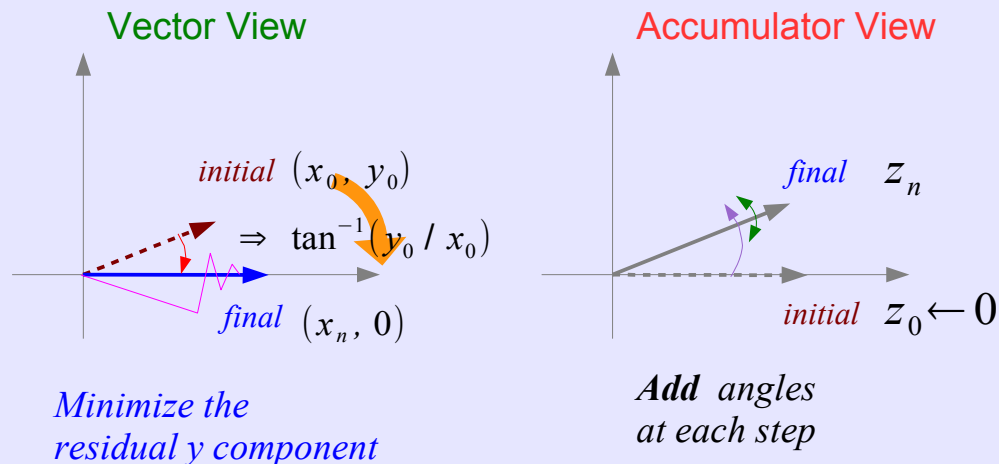
$$z_0 \leftarrow 0$$

$$z_n \rightarrow z_0 + \tan^{-1}(y_0/x_0)$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < 0$$

$$d_i = -1 \quad \text{otherwise}$$



Input

$$(x_0, y_0) \rightarrow \text{ratio } \frac{y_0}{x_0}$$

$$(0, y_0) \rightarrow \text{ratio } \pm\infty$$

Output

Angle Accumulator Value

→ CORDIC gain does not affect

$$x_n = z_0 + \tan^{-1}(y_0/x_0)$$

E. Vector Magnitude

Vectoring Mode

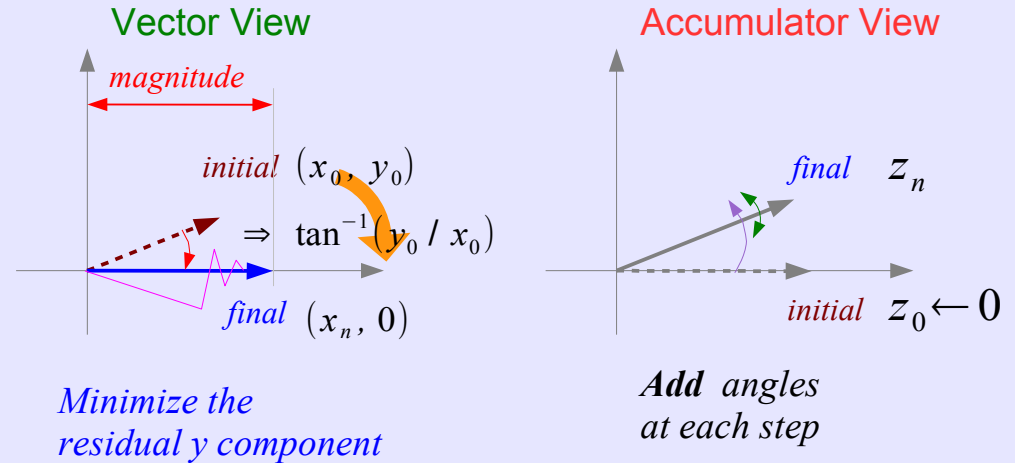
$$z_0 \leftarrow 0$$

$$z_n \rightarrow z_0 + \tan^{-1}(y_0/x_0)$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < 0$$

$$d_i = -1 \quad \text{otherwise}$$



The magnitude:

- byproduct of computing arctangent
- the result vector is aligned with x-axis
- the x component of the result vector
- increased by CORDIC gain
- can be scaled by the processor gain
- one MULT hardware cost

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

The accuracy of the magnitude result

- Improves by 2 bits for each iteration performed

F. Cartesian to Polar Transformation

Vectoring Mode

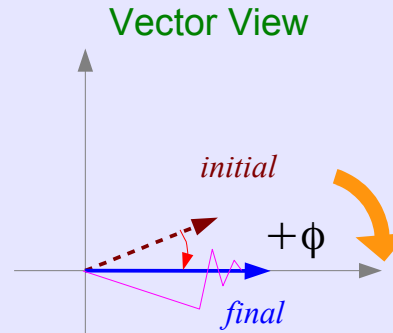
$$z_0 \leftarrow 0$$

$$z_n \rightarrow z_0 + \tan^{-1}(y_0/x_0)$$

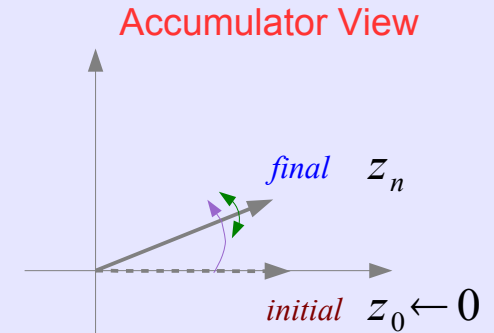
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < 0$$

$$d_i = -1 \quad \text{otherwise}$$



Minimize the residual y component



Add angles at each step

input vector (x, y)

magnitude $r = \sqrt{x^2 + y^2}$



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

phase angle $\phi = \tan^{-1}(y/x)$



$$z_n = z_0 + \tan^{-1}(y_0/x_0)$$

G. ArcSine (1)

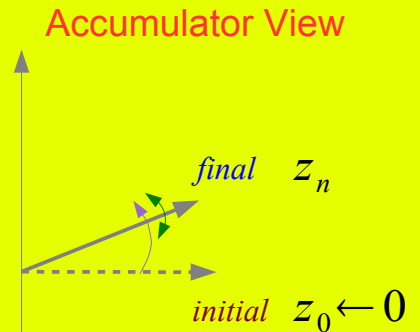
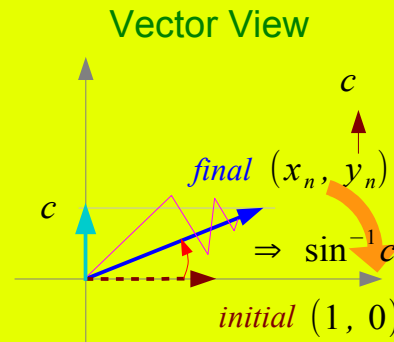
Exploit Vector Mode HW

$$z_n \rightarrow z_0 + \sin^{-1}\left(\frac{c}{A_n \cdot x_0}\right)$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < c$$

$$d_i = -1 \quad \text{otherwise}$$



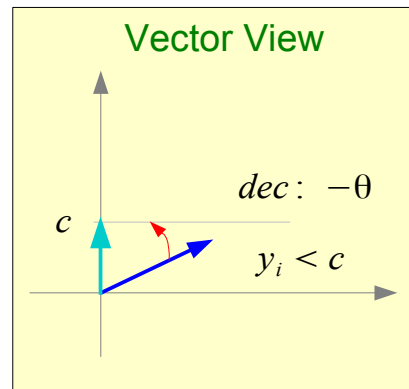
$$x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i}$$

$$y_{i+1} = y_i + x_i \cdot d_i \cdot 2^{-i}$$

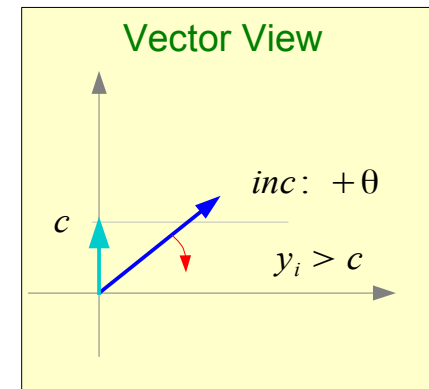
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < c$$

$$d_i = -1 \quad \text{otherwise}$$



y_i < c Dec Angle
Add (-) Angle



y_i > c Inc Angle
Add (+) Angle

G. ArcSine (2)

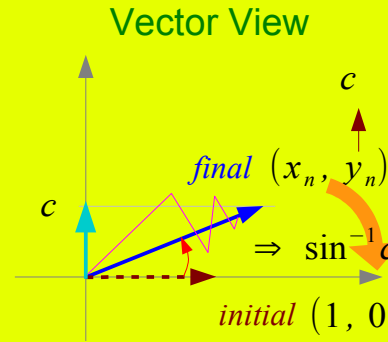
Exploit Vector Mode HW

$$z_n \rightarrow z_0 + \sin^{-1}\left(\frac{c}{A_n \cdot x_0}\right)$$

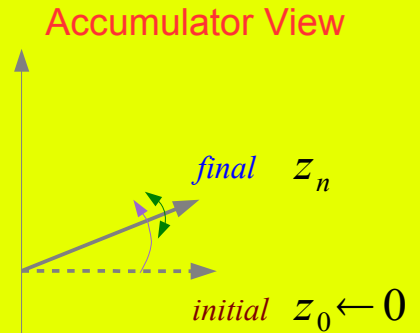
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < c$$

$$d_i = -1 \quad \text{otherwise}$$



Minimize the residual y component



Add angles at each step

$$x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i}$$

$$y_{i+1} = y_i + x_i \cdot d_i \cdot 2^{-i}$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < c$$

$$d_i = -1 \quad \text{otherwise}$$

$$x_n = \sqrt{(A_n \cdot x_0)^2 - c^2}$$

$$y_n = c$$

$$z_n = z_0 + \sin^{-1}\left(\frac{c}{A_n \cdot x_0}\right)$$

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

H. Arccosine (1)

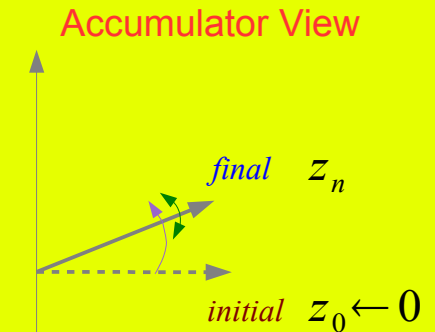
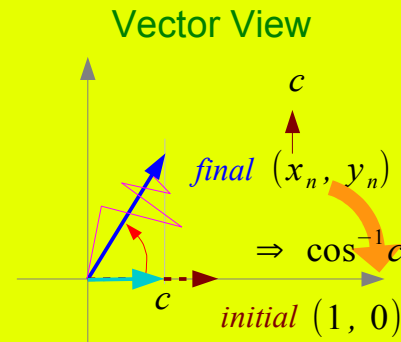
Exploit Vector Mode HW

$$z_n \rightarrow z_0 + \sin^{-1}\left(\frac{c}{A_n \cdot x_0}\right)$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < c$$

$$d_i = -1 \quad \text{otherwise}$$



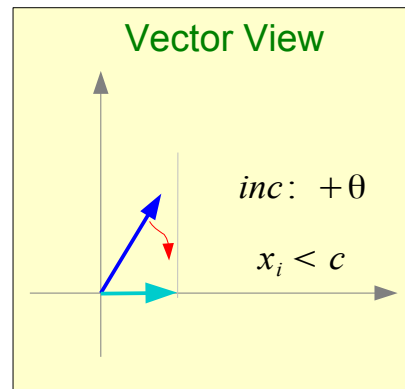
$$x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i}$$

$$y_{i+1} = y_i + x_i \cdot d_i \cdot 2^{-i}$$

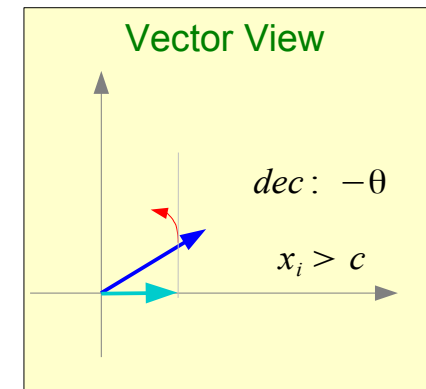
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } x_i > c$$

$$d_i = -1 \quad \text{otherwise}$$



$x_i < c$ Inc Angle
Add (+) Angle



$x_i > c$ Dec Angle
Add (-) Angle

H. Arccosine (1)

Exploit Vector Mode HW

$$z_n \rightarrow z_0 + \sin^{-1}\left(\frac{c}{A_n \cdot x_0}\right)$$

$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } y_i < c$$

$$d_i = -1 \quad \text{otherwise}$$

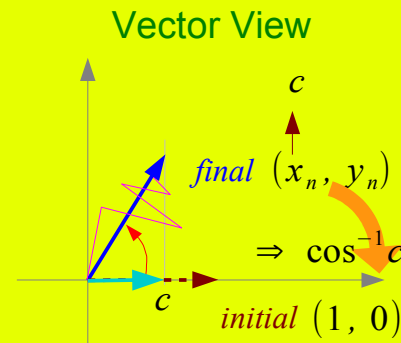
$$x_{i+1} = x_i - y_i \cdot d_i \cdot 2^{-i}$$

$$y_{i+1} = y_i + x_i \cdot d_i \cdot 2^{-i}$$

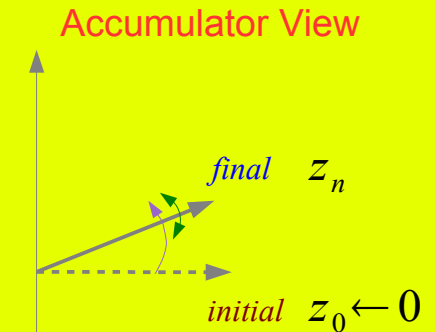
$$z_{i+1} = z_i - d_i \cdot \tan^{-1}(2^{-i})$$

$$d_i = +1 \quad \text{if } x_i > c$$

$$d_i = -1 \quad \text{otherwise}$$



Minimize the residual x component



Add angles at each step

$$y_n = \sqrt{(A_n \cdot y_0)^2 - c^2}$$

$$x_n = c$$

$$z_n = z_0 + \cos^{-1}\left(\frac{c}{A_n \cdot y_0}\right)$$

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

References

- [1] <http://en.wikipedia.org/>
- [2] CORDIC FAQ, www.dspguru.com