Angle Recoding 2. Wu 3. MVR

20180925 Tue

② MVR (Modified Vector Rotational)

two modifications

① repeatition of elementary angles

each micro-rotation of elementary angle
can be performed repeatedly

- more possible combinations
- smaller ξ_m

② confinement of total micro-rotation number

confine the iteration number
in the micro-rotation phase
to R_m ($R_m \ll W$)

The role of R_m is quite similar
to the number of non-zero digit
N_D in CSD recoding scheme

MVR * repeatition of elementary angles

$\theta = a(2) * 2 = \tan^{-1}(2^{-2}) * 2$

AR $+a(1) + a(5) - a(8) - a(10) - a(14)$

```
k= 0 theta= 0.4899573 ik= 1 uik=+1 a[ 1]= 0.4636476, new theta= 0.0263097
k= 1 theta= 0.0263097 ik= 5 uik=+1 a[ 5]= 0.0312398, new theta=-0.0049301
k= 2 theta=-0.0049301 ik= 8 uik=-1 a[ 8]= 0.0039062, new theta=-0.0010239
k= 3 theta=-0.0010239 ik=10 uik=-1 a[10]= 0.0009766, new theta=-0.0000473
k= 4 theta=-0.0000473 ik=14 uik=-1 a[14]= 0.0000610, new theta= 0.0000137
```

Conventional CORDIC $+a(0) - a(1) + a(2) - a(3) + a(4) - a(5) + a(6) + a(7) - a(8) - a(9) - a(10) - a(11) + a(12) + a(13) - a(14) - a(15)$

```
k= 0 theta= 0.4899573 u[ 0]=+1 a[ 0]= 0.7853982, new theta=-0.2954408
k= 1 theta=-0.2954408 u[ 1]=-1 a[ 1]= 0.4636476, new theta= 0.1682068
k= 2 theta= 0.1682068 u[ 2]=+1 a[ 2]= 0.2449787, new theta=-0.0767719
k= 3 theta=-0.0767719 u[ 3]=-1 a[ 3]= 0.1243550, new theta= 0.0475831
k= 4 theta= 0.0475831 u[ 4]=+1 a[ 4]= 0.0624188, new theta=-0.0148357
k= 5 theta=-0.0148357 u[ 5]=-1 a[ 5]= 0.0312398, new theta= 0.0164041
k= 6 theta= 0.0164041 u[ 6]=+1 a[ 6]= 0.0156237, new theta= 0.0007804
k= 7 theta= 0.0007804 u[ 7]=+1 a[ 7]= 0.0078123, new theta=-0.0070319
k= 8 theta=-0.0070319 u[ 8]=-1 a[ 8]= 0.0039062, new theta=-0.0031257
k= 9 theta=-0.0031257 u[ 9]=-1 a[ 9]= 0.0019531, new theta=-0.0011726
k=10 theta=-0.0011726 u[10]=-1 a[10]= 0.0009766, new theta=-0.0001960
k=11 theta=-0.0001960 u[11]=-1 a[11]= 0.0004883, new theta= 0.0002923
k=12 theta= 0.0002923 u[12]=+1 a[12]= 0.0002441, new theta= 0.0000481
k=13 theta= 0.0000481 u[13]=+1 a[13]= 0.0001221, new theta=-0.0000740
k=14 theta=-0.0000740 u[14]=-1 a[14]= 0.0000610, new theta=-0.0000129
k=15 theta=-0.0000129 u[15]=-1 a[15]= 0.0000305, new theta= 0.0000176
```

data : W-bit word length

the iteration number : N $\quad N \leqslant W$

the restricted iteration number : R_m $\quad R_m \ll W$

$i \in [0, R_m - 1] \quad R_m < W$

the angle quantization error

$$\xi_{m,MVR} \triangleq \theta - \sum_{i=0}^{R_m-1} \alpha(i)\, a(s(i))$$

the rotational sequence

$s(i) \in \{0, 1, \cdots, W-1\}$ * repetition allowed

the micro-rotation angle in the i-th iteration

the directional sequence

$\alpha(i) \in \{-1, 0, +1\}$

the direction of the i-th micro-rotation of $a(s(i))$

$$\alpha(i)\, a(s(i)) = \tilde{\theta}(i)$$

the rotational sequence

$s(i) \in \{0, 1, \cdots, W-1\}$ [0, 3, 6, 7]

the directional sequence

$\alpha(i) \in \{-1, 0, +1\}$ [1, -1, -1, 1]

atan(2^0) - atan(2^{-3}) - atan(2^{-6}) + atan(2^{-7})

$$\alpha(i)\, a(s(i)) = \tilde{\theta}(j)$$

MVR-CORDIC Algorithm with $R_m = 4$	Greedy Algorithm	3	$\overline{\alpha} = [1\ \ -1\ \ -1\ \ -1]$ $\overline{s} = [0\ \ 3\ \ 6\ \ 7]$	$5.2891 * 10^{-4}$
	Semi-greedy Algorithm ($D = 2$)	4	$\overline{\alpha} = [1\ \ -1\ \ -1\ \ 1]$ $\overline{s} = [0\ \ 3\ \ 5\ \ 7]$	$5.2033 * 10^{-4}$
	TBS Algorithm	5	$\overline{\alpha} = [1\ \ 1\ \ -1\ \ -1]$ $\overline{s} = [1\ \ 2\ \ 4\ \ 7]$	$2.5911 * 10^{-4}$

```
>> s = [0, 3, 6, 7]
>> alpha = [1, -1, -1, -1]
>> sum(atan(2.^(-s) .* alpha))
ans =  0.63761
>>
```

```
>> s = [0, 3, 5, 7]
>> alpha = [1, -1, -1, 1]
>> sum(atan(2.^(-s) .* alpha))
ans =  0.63762
```

-

```
>> alpha = [1, 1, -1, -1]
>> s = [1, 2, 4, 7]
>> sum(atan(2.^(-s) .* alpha))
ans =  0.63840
```

0	7.85398163397448e-01
1	4.63647609000806e-01
2	2.44978663126864e-01
3	1.24354994546761e-01
4	6.24188099959574e-02
5	3.12398334302683e-02
6	1.56237286204768e-02
7	7.81234106010111e-03
8	3.90623013196697e-03
9	1.95312251647882e-03
10	9.76562189559319e-04
11	4.88281211194898e-04
12	2.44140620149362e-04
13	1.22070311893670e-04
14	6.10351561742088e-05
15	3.05175781155261e-05

W=16

0	7.85398163397448e-01	s(0)
1	4.63647609000806e-01	.
2	2.44978663126864e-01	.
3	1.24354994546761e-01	s(1)
4	6.24188099959574e-02	.
5	3.12398334302683e-02	.
6	1.56237286204768e-02	s(2)
7	7.81234106010111e-03	s(3)
8	3.90623013196697e-03	.
9	1.95312251647882e-03	.
10	9.76562189559319e-04	.
11	4.88281211194898e-04	.
12	2.44140620149362e-04	.
13	1.22070311893670e-04	.
14	6.10351561742088e-05	.
15	3.05175781155261e-05	.

0	7.85398163397448e-01	s(0)
3	1.24354994546761e-01	s(1)
6	1.56237286204768e-02	s(2)
7	7.81234106010111e-03	s(3)

Rm=4

s(0)=0
s(1)=3
s(2)=6
s(3)=7

0	7.85398163397448e-01
1	4.63647609000806e-01
2	2.44978663126864e-01
3	1.24354994546761e-01
4	6.24188099959574e-02
5	3.12398334302683e-02
6	1.56237286204768e-02
7	7.81234106010111e-03
8	3.90623013196697e-03
9	1.95312251647882e-03
10	9.76562189559319e-04
11	4.88281211194898e-04
12	2.44140620149362e-04
13	1.22070311893670e-04
14	6.10351561742088e-05
15	3.05175781155261e-05

AQ & MVR CORDIC

$$\xi_{m,MVR} \triangleq \theta - \left[\sum_{j=0}^{R_m-1} \alpha(j)\, a(s(j)) \right]$$

the rotational sequence $s(j)$

$j = 0, 1, 2, \cdots, R_m - 1$

↓

$s(j) \in \{0, 1, \cdots, W-1\}$ rotational sequence

determines the micro-rotation angle $a(s(j))$ in the j-th iteration

the directional sequence $\alpha(j)$

$\alpha(j) \in \{-1, 0, +1\}$

controls the direction of the j-th micro-rotation of $a(s(j))$

$$\alpha(j)\, a(s(j)) = \tilde{\theta}(j)$$

$i =$	0,	1,	2,	3,	$\cdots$,	$W-1$		
$s(j) =$	0,	1,	2,	3,	$\cdots$,	$W-1$		rotational sequence
$\alpha(j) =$	-1,	0,	0,	$+1$,	$\cdots$,	-1		directional sequence
$j =$	0,	–,	–,	1,	$\cdots$,	R_m-1		effective iteration number

$R_m \ll W$

i	j	S(j)	
(0)	0	S(0) = 0	
1			
2			
3			
(4)	1, 4	S(1) = 4 , S(4) = 4	repetition allowed
(5)	2	S(2) = 5	
6			
7			
(8)	3	S(3) = 8	
9		rotational sequence	
10	effective iteration number		
11			
12			
13			
14			
W-1 = 15			

i	Conventional CORDIC	j	S(j)
(0)	S(0) = 0	0	S(0) = 0
1	S(1) = 1		
2	S(2) = 2		
3	S(3) ··3		
(4)	S(4) = 4	1	S(1) = 4
(5)	S(5) = 5	2	S(2) = 5
6	S(6) = 6		
7	S(7) = 7		
(8)	S(8) = 8	3	S(3) = 8
9	S(9) = 9		
10	S(10) = 10		
11	S(11) = 11		
12	S(12) = 12		
13	S(13) = 13		
14	S(14) = 14		
15	S(15) = 15		

effective iteration number

rotational sequence

W-1 =

sub-angle $(\alpha(j)\, a(s(j)))$ ~ $\tilde{\theta}(j)$

$$\xi_{m,AR} = \theta - \left[\sum_{j=0}^{N'-1} \tilde{\theta}(j) \right], \quad \tilde{\theta}(j) = \tan^{-1}(\alpha(j) \cdot 2^{-s(j)})$$

$$= \theta - \left[\sum_{j=0}^{N'-1} \tan^{-1}(\alpha(j) \cdot 2^{-s(j)}) \right]$$

$N' \triangleq \sum_{j=0}^{N-1} |\mu(j)|$ the effective iteration number

EAS formed by MVR-CORDIC
is the same as AR
also performs AQ

the EAS consists of all possible values of $\tilde{\theta}(j)$

the EAS S_1 in AR

$$S_1 = \{ \tan^{-1}(\alpha^* \cdot 2^{-s^*}) : \alpha^* \in \{-1, 0, +1\}, \ s^* \in \{0, 1, \cdots, N-1\} \}$$

The major difference

1) the total number of sub-angles N_A

the total iteration number

in the micro-rotation phase

is kept fixed to a pre-defined value of R_m

$N_A = R_m$

2) the sub-angle θ_i corresponds to $\alpha(j)\,a(s(j))$

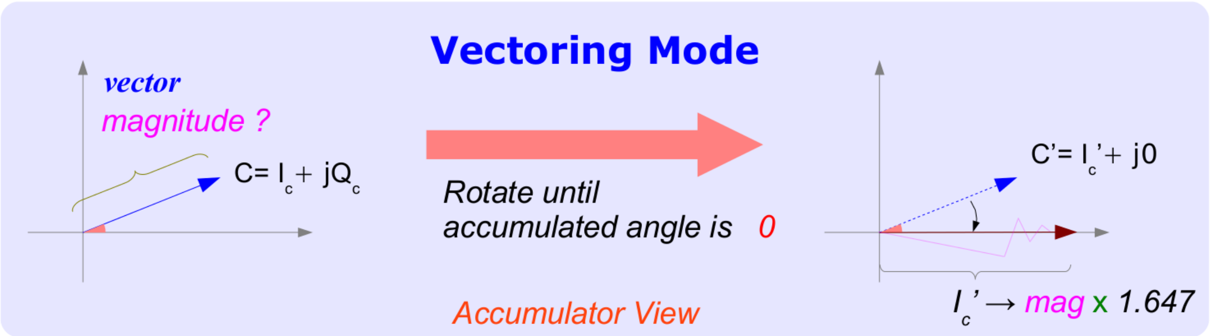
$$\theta_j = \alpha(j)\,a(s(j)) = \tilde{\theta}_j$$

MVR (Modified Vector Rotation)

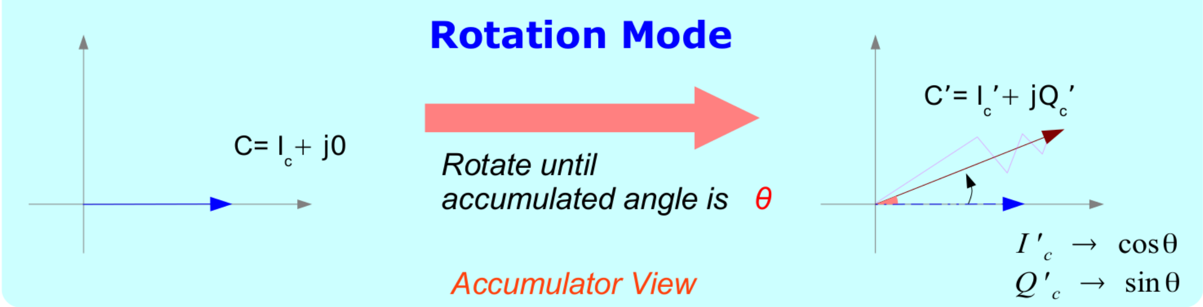
1) Repeat of Elementary Angles θ_i, θ_i

2) fixed total micro-rotation Number R_m

* Vector Rotation Mode
* and the rotation angles are known in advance



vector rotation mode



Modified Vector Rotational MVR CORDIC

- reduce the iteration number
- maintaining the SQNR performance
- modifying the basic microrotation procedure

Three Searching Algorithm

1. the selective prerotation
2. the selective scaling
3. iteration-tradeoff scheme

Optimization Problem

EAS point of view

Given θ, find the combination of Rm elementary angles from EAS S_1, such that the angle quantization error $|\xi_{m,MUR}|$ is minimized.

semi-greedy algorithm
trade offs between computational complexities and performance

key issue in the MVR-CORDIC
is to find the best sequences of
$s(i)$ and $\alpha(i)$ to minimize $|\xi_m|$
subject to the constraint that
the total iteration number is confined to R_m

1) Greedy Algorithm
2) Exhaustive Algorithm
3) Semigreedy Algorithm

data : W-bit word length

the iteration number : N $\quad N \leqslant W$

the restricted iteration number : $R_m \quad R_m \ll W$

Hu's greedy algorithm

$\theta(0) = \theta$, $\{\mu(i) = 0,\ 0 \leq i \leq N-1\}$, $k=0$.

repeat until $|\theta(k)| < a(N-1)$ Do

Choose i_k, $0 \leq i_k \leq N-1$

$$\big|\,|\theta(k)| - a(i_k)\big| = \min_{0 \leq i \leq N-1} \big|\,|\theta(k)| - a(i_k)\big|$$

$$\theta(k+1) = \theta(k) - \mu(i_k)\, a(i_k)$$

$$\mu(i_k) = \text{Sign}(\theta(k))$$

$J(i) = |\theta(i) - \alpha(i)\, a(s(i))|$ is minimited

1) Greedy Algorithm

given θ, W, R_m

try to approach the target rotation angle, θ, step by step
in each step, decisions are made on $\alpha(i)$ and $s(i)$
by choosing the best combination of $\alpha(i)\, a(s(i))$
so as to minimize $|\xi_m|$

$\alpha(i)$ and $s(i)$ are determined such that
the error function $J(i) = |\theta(i) - \alpha(i)\, a(s(i))|$ is minimized

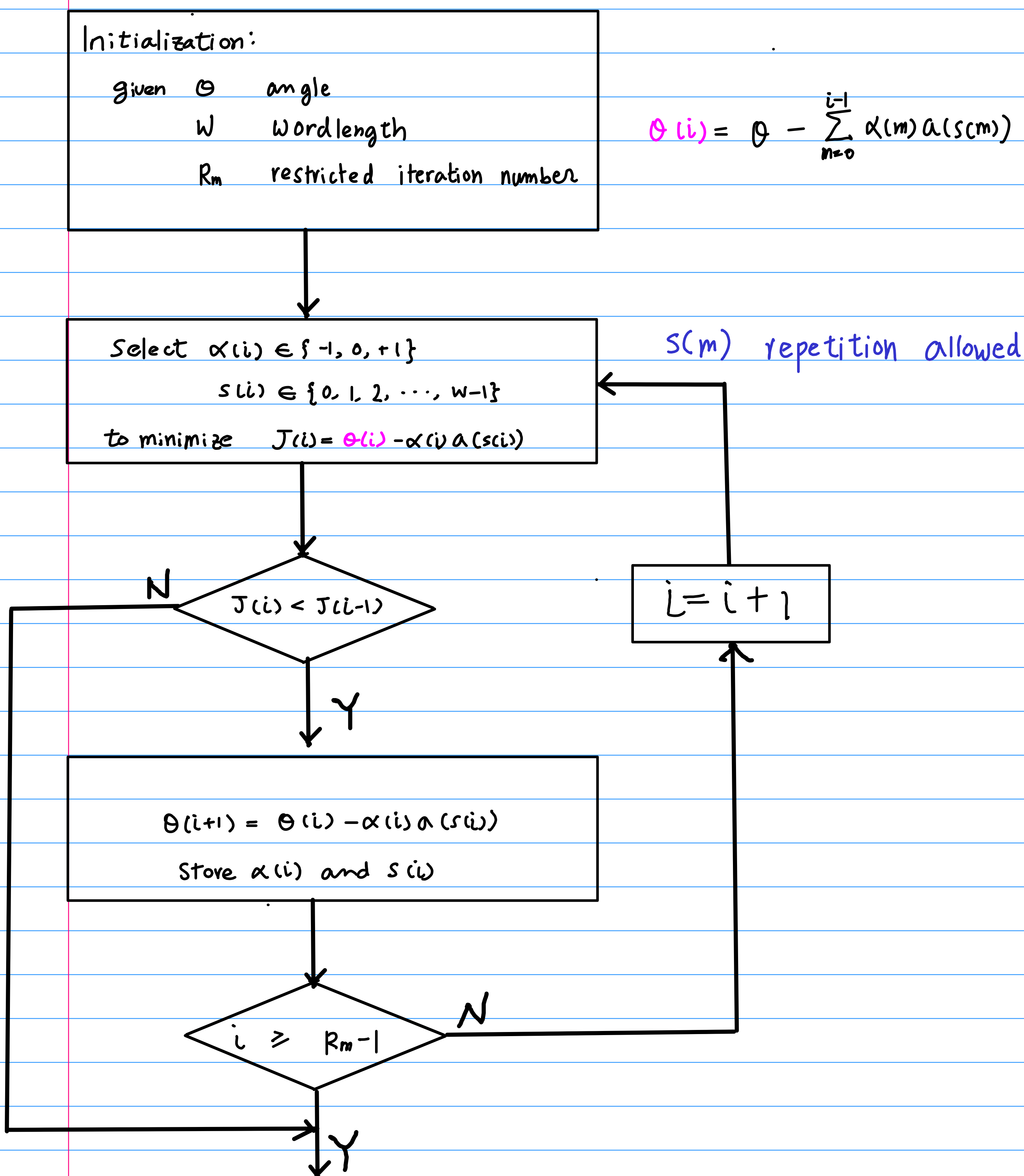
$\theta(i)$: the residue angle in the i-th step

$$\theta(i) = \theta - \sum_{m=0}^{i-1} \alpha(m)\, a(s(m))$$

the searching is terminated
- if no further improvements can be found
  $J(i) \geq J(i-1)$
- if the iteration i reaches $R_m - 1$

$\alpha(R_m-1)$ and $s(R_m-1)$ are determined
at the end of the searching

the greedy algorithm terminates
only when the residue angle error
cannot be further reduced.



Initialization:
given Θ angle
W wordlength
Rm restricted iteration number
θ(i) = θ − Σ_{m=0}^{i-1} α(m) a(s(m))
Select α(i) ∈ {-1, 0, +1}
s(i) ∈ {0, 1, 2, ···, W−1}
to minimize J(i) = θ(i) − α(i) a(s(i))
s(m) repetition allowed
J(i) < J(i-1)
N
Y
i = i + 1
θ(i+1) = θ(i) − α(i) a(s(i))
Store α(i) and s(i)
i ≥ Rm − 1
N
Y

2) Exhaustive Algorithm

search for the entire solution space

$$\underset{3}{\overset{\alpha(i)}{\{-1, 0, +1\}}} \times \underset{W}{\overset{a(s(i))}{\{s(0), s(1), \cdots, s(W-1)\}}} \times \underset{R_m}{\overset{i}{\{0, 1, \cdots, R_m-1\}}} \Rightarrow (3W)^{R_m}$$

all possible combinations of

$$\sum_{i=0}^{R_m-1} \alpha(i)\, a(s(i))$$

in a <u>single</u> step

decisions for $\alpha(i)$ and $s(i)$, $0 \leq i \leq R_m - 1$

by minimizing the error function

$$J = \left| \theta - \sum_{i=0}^{R_m-1} \alpha(i)\, a(s(i)) \right|$$

global optimal solution

Initialization:

given Θ, W, R_m

let $\Theta(0) = \Theta$,

$i = 0$

$J(-1) = \infty$

↓

Select $\alpha(i) \in \{-1, 0, +1\}$

$s(i) \in \{0, 1, 2, \cdots, W-1\}$

for $0 \leq i \leq R_m - 1$

to minimize $J(i) = \Theta - \sum_{i=0}^{R_m-1} \alpha(i)\, a(s(i))$

↓

Store $\alpha(i)$ and $s(i)$

for $0 \leq i \leq R_m - 1$

$(3 \cdot W) \cdot (3 \cdot W) \cdots (3 \cdot W)$

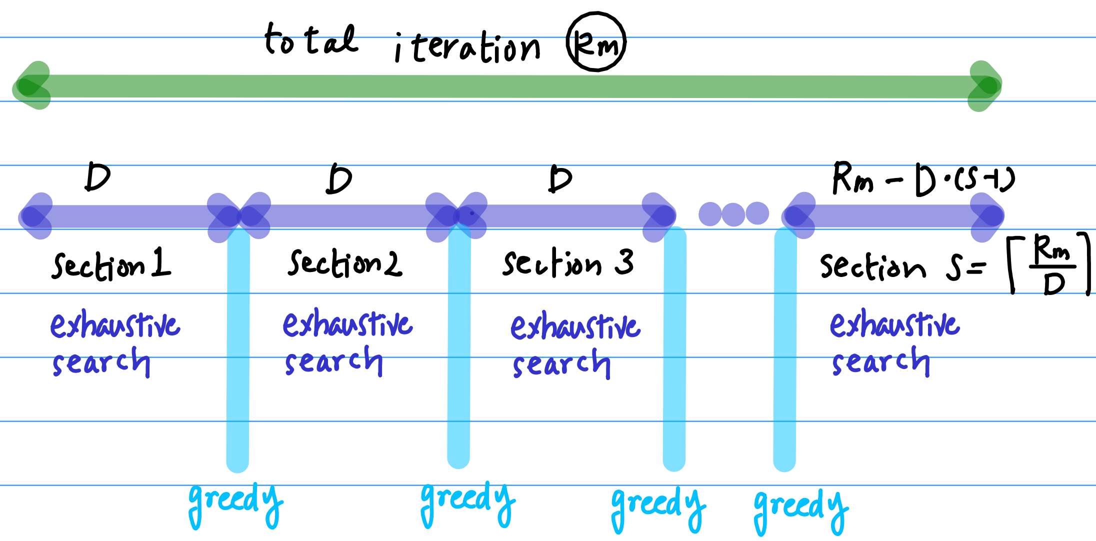
$= 3^{R_m} \cdot W^{R_m}$

3) Semi-greedy Algorithm

a combination of greedy and exhaustive algorithm

the search space of $\alpha(i)$ and $S(i)$ for $0 \leq i \leq R_m - 1$
are divided into several sections
with D iterations as a segment
D iterations ↓ block length; segment ↓ block

the segmentation scheme



in the i-th block

decision of $\alpha(k)$ and $s(k)$ for $iD \leq k \leq (i+1)D-1$

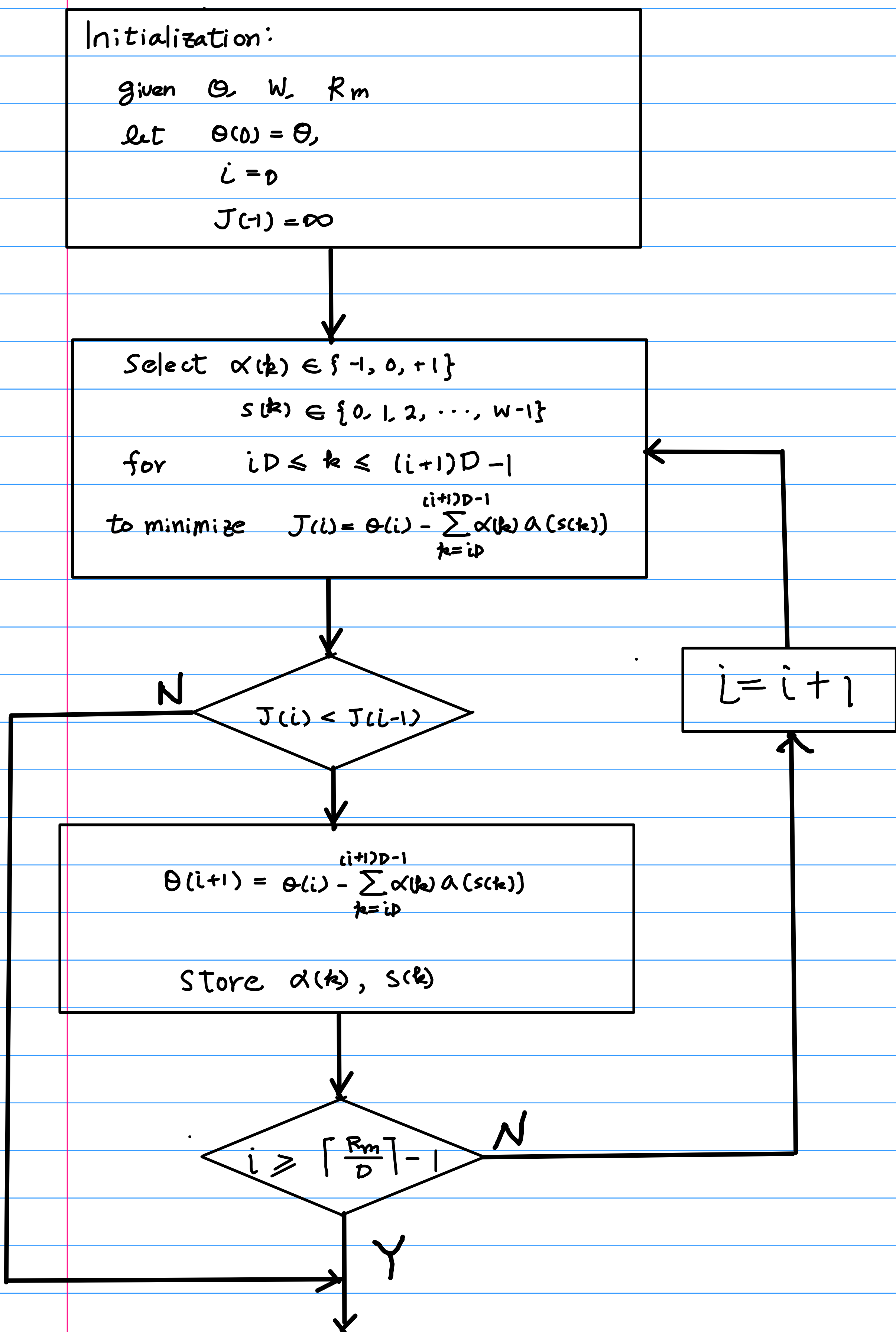
minimizes $$J = \left| \theta(i) - \sum_{k=iD}^{(i+1)D-1} \alpha(k)\, a(s(k)) \right|$$

where $$\theta(i) = \theta - \sum_{m=0}^{i-1} \left[\sum_{k=mD}^{(m+1)D-1} \alpha(k)\, a(s(k)) \right]$$

the residue angle in the i-th step

$$S = \left\lceil \frac{R_m}{D} \right\rceil$$

$$\theta(i) = \theta - \left[\sum_{k=0 \cdot D}^{1 \cdot D-1} \alpha(k)\, a(s(k)) + \sum_{k=1D}^{2 \cdot D-1} \alpha(k)\, a(s(k)) + \cdots + \sum_{k=2D}^{S \cdot D-1} \alpha(k)\, a(s(k)) \right]$$


Initialization:
given Θ, W, Rm
let Θ(0) = Θ,
i = 0
J(-1) = ∞
Select α(k) ∈ {-1, 0, +1}
s(k) ∈ {0, 1, 2, ···, W-1}
for iD ≤ k ≤ (i+1)D -1
to minimize $J(i) = \Theta(i) - \sum_{k=iD}^{(i+1)D-1} \alpha(k)\, a(s(k))$
N
J(i) < J(i-1)
$\Theta(i+1) = \Theta(i) - \sum_{k=iD}^{(i+1)D-1} \alpha(k)\, a(s(k))$
Store α(k), s(k)
$i \geq \lceil \frac{R_m}{D} \rceil - 1$
N
Y
i = i + 1

```
#include <stdio.h>
#include <math.h>

#define N 16
#define Rm 8

//-----------------------------------------------------------------
// conventional cordic
// input initial angle  : angle[0]
// ouput residue angles : angle[1] .. angle[N]
//-----------------------------------------------------------------
void conventional_cordic(double a[], double angle[]) {
  int k = 0, uk;

  printf("* conventional cordic ... \n");

  for (k=0; k<N; ++k) {
    uk = (angle[k] >= 0) ? +1 : -1;

    printf("k=%2d angle[%2d]=%10.7f ", k, k, angle[k]);
    printf("uk=%+d ", uk);

    angle[k+1] = angle[k] - uk * a[k];

    printf("a[%2d]=%10.7f ", k, a[k]);
    printf("angle[%2d]=%10.7f \n", k, angle[k+1]);
  }
}


int find_multiple_angle(double a[], double angle[], int k) {
  int m, mm;
  double residue, minval;

  mm = -1;
  minval = 1e+100;
  for (m=1; m+k<N; ++m) {
    if (angle[k] > 0) residue = angle[k] - (m+1)*a[k+m];
    else              residue = angle[k] + (m+1)*a[k+m];

    if (fabs(angle[k+m+1]) >= fabs(residue)) {
      if (minval >= fabs(residue)) {
        minval = fabs(residue);
        mm = m;
      }
    }
  }
  return mm;
}
```

```
int main(void) {
  double a[N], angle[N+1];
  double theta; // = 4*atan(pow(2,-5));

  int uk;
  int k, i, s;
  int m, km;

  for (i=0; i<N; ++i) {
    a[i] = atan(1./pow(2, i));
  }


for (s=2; s<3; ++s) {

  angle[0] = theta = s * atan(pow(2, -6));

  printf("theta= %2d * atan(pow(2,-6) = %10g \n", s, theta);

  conventional_cordic(a, angle);

  for (k=0; k<N; ++k) {
    uk = (theta >= 0) ? +1 : -1;

    m = find_multiple_angle(a, angle, k);

    if (m > 0) {
      km = k+m;
      for (i=0; i<=m; i++) {
        printf("k=%2d theta=%10.7f ", k, theta);
        printf("i=%2d uk=%+d m=%2d ", i, uk, m-i);

        theta = theta - uk * a[km];
        k++;

        printf("a[%2d]=%10.7f, new theta=%10.7g \n", km, a[km], theta);
      }
      k--;
    } else {
        printf("k=%2d theta=%10.7f ", k, theta);
        printf("i=%2d uk=%+d m=%2d ", 0, uk, m);

        theta = theta - uk * a[k];

        printf("a[%2d]=%10.7f, new theta=%10.7g \n", k, a[k], theta);
    }
  }
}

}
```

```
theta=  2 * atan(pow(2,-6) =  0.0312475
* conventional cordic ...
k= 0 angle[ 0]= 0.0312475 uk=+1 a[ 0]= 0.7853982 angle[ 0]=-0.7541507
k= 1 angle[ 1]=-0.7541507 uk=-1 a[ 1]= 0.4636476 angle[ 1]=-0.2905031
k= 2 angle[ 2]=-0.2905031 uk=-1 a[ 2]= 0.2449787 angle[ 2]=-0.0455244
k= 3 angle[ 3]=-0.0455244 uk=-1 a[ 3]= 0.1243550 angle[ 3]= 0.0788306
k= 4 angle[ 4]= 0.0788306 uk=+1 a[ 4]= 0.0624188 angle[ 4]= 0.0164118
k= 5 angle[ 5]= 0.0164118 uk=+1 a[ 5]= 0.0312398 angle[ 5]=-0.0148281
k= 6 angle[ 6]=-0.0148281 uk=-1 a[ 6]= 0.0156237 angle[ 6]= 0.0007956
k= 7 angle[ 7]= 0.0007956 uk=+1 a[ 7]= 0.0078123 angle[ 7]=-0.0070167
k= 8 angle[ 8]=-0.0070167 uk=-1 a[ 8]= 0.0039062 angle[ 8]=-0.0031105
k= 9 angle[ 9]=-0.0031105 uk=-1 a[ 9]= 0.0019531 angle[ 9]=-0.0011573
k=10 angle[10]=-0.0011573 uk=-1 a[10]= 0.0009766 angle[10]=-0.0001808
k=11 angle[11]=-0.0001808 uk=-1 a[11]= 0.0004883 angle[11]= 0.0003075
k=12 angle[12]= 0.0003075 uk=+1 a[12]= 0.0002441 angle[12]= 0.0000634
k=13 angle[13]= 0.0000634 uk=+1 a[13]= 0.0001221 angle[13]=-0.0000587
k=14 angle[14]=-0.0000587 uk=-1 a[14]= 0.0000610 angle[14]= 0.0000023
k=15 angle[15]= 0.0000023 uk=+1 a[15]= 0.0000305 angle[15]=-0.0000282
k= 0 theta= 0.0312475 i= 0 uk=+1 m=-1 a[ 0]= 0.7853982, new theta=-0.7541507
k= 1 theta=-0.7541507 i= 0 uk=-1 m=-1 a[ 1]= 0.4636476, new theta=-0.2905031
k= 2 theta=-0.2905031 i= 0 uk=-1 m= 1 a[ 3]= 0.1243550, new theta=-0.1661481
k= 3 theta=-0.1661481 i= 1 uk=-1 m= 0 a[ 3]= 0.1243550, new theta=-0.04179311
k= 4 theta=-0.0417931 i= 0 uk=-1 m=-1 a[ 4]= 0.0624188, new theta= 0.0206257
k= 5 theta= 0.0206257 i= 0 uk=+1 m= 3 a[ 8]= 0.0039062, new theta=0.01671947
k= 6 theta= 0.0167195 i= 1 uk=+1 m= 2 a[ 8]= 0.0039062, new theta=0.01281324
k= 7 theta= 0.0128132 i= 2 uk=+1 m= 1 a[ 8]= 0.0039062, new theta=0.008907012
k= 8 theta= 0.0089070 i= 3 uk=+1 m= 0 a[ 8]= 0.0039062, new theta=0.005000781
k= 9 theta= 0.0050008 i= 0 uk=+1 m=-1 a[ 9]= 0.0019531, new theta=0.003047659
k=10 theta= 0.0030477 i= 0 uk=+1 m= 1 a[11]= 0.0004883, new theta=0.002559378
k=11 theta= 0.0025594 i= 1 uk=+1 m= 0 a[11]= 0.0004883, new theta=0.002071096
k=12 theta= 0.0020711 i= 0 uk=+1 m=-1 a[12]= 0.0002441, new theta=0.001826956
k=13 theta= 0.0018270 i= 0 uk=+1 m=-1 a[13]= 0.0001221, new theta=0.001704886
k=14 theta= 0.0017049 i= 0 uk=+1 m= 1 a[15]= 0.0000305, new theta=0.001674368
k=15 theta= 0.0016744 i= 1 uk=+1 m= 0 a[15]= 0.0000305, new theta=0.00164385
```