Digital Differential Analyzer (DDA)

In Scan Conversion which is y= mx+c , Has two major problems, which are 1. Multiplication 2. Floating Point. Multiplication increase time complexity and in terms of floating point pixels cant be a float point, it always has to be integer so, we need to roundoff the float points which also increase the time complexity and also to some extend it cannot preserve the image shape due to roundoff.

Advantage : In DDA, It overcome the multiplication problem with addtion that reduce the time complexity

Disadvantage : In DDA, we also use the slope m in our calculation, In most of the cases m is a floating point. Hence, floating and round of issue still remains in DDA.

DDA Algorithm (Revised Version)

If -1<m<1 then,

If $x_1 > x_0$,

$$x_{k+1} = x_k + 1$$
$$y_{k+1} = y_k + m$$

Else,

$$x_{k+1} = x_k - 1$$
$$y_{k+1} = y_k - m$$

Then roundoff y.

Otherwise, if -∞<m<-1 or 1<m< ∞ then,

If $y_1 > y_0$,

$$y_{k+1} = y_k + 1$$
$$x_{k+1} = x_k + \frac{1}{m}$$

Else,

$$y_{k+1} = y_k - 1$$
$$x_{k+1} = x_k - \frac{1}{m}$$

Then roundoff x.

If $(-1<m<1)$

If $x_1 > x_0$

$$x_{k+1} = x_k + 1$$
$$y_{k+1} = y_k + m$$

If $x_0 > x_1$

$$x_{k+1} = x_k - 1$$
$$y_{k+1} = y_k - m$$

If $(1<m<\infty)$ $(-\infty<m<-1)$

If $y_1 > y_0$

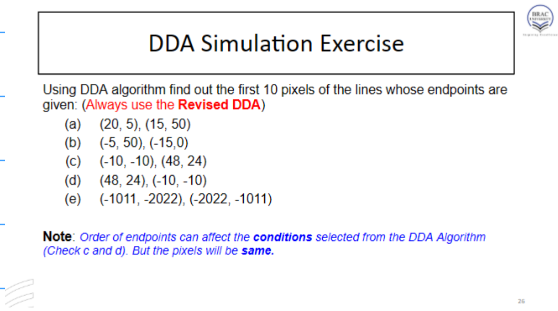
$$x_{k+1} = x_k + \frac{1}{m}$$
$$y_{k+1} = y_k + 1$$

If $y_0 > y_1$

$$x_{k+1} = x_k - \frac{1}{m}$$
$$y_{k+1} = y_k - 1$$

Step 1: First determine m, then check if m is $(-1<m<1)$ if yes then check x. otherwise, check y

step 2: check if $x1 > x0$ or for 2nd condition $y1>y0$



DDA Simulation Exercise

Using DDA algorithm find out the first 10 pixels of the lines whose endpoints are given: (Always use the **Revised DDA**)

(a) (20, 5), (15, 50)
(b) (-5, 50), (-15,0)
(c) (-10, -10), (48, 24)
(d) (48, 24), (-10, -10)
(e) (-1011, -2022), (-2022, -1011)

Note: *Order of endpoints can affect the **conditions** selected from the DDA Algorithm (Check c and d). But the pixels will be **same**.*

26

a. (20, 5), (15, 50)

Steps	X	Round(x)	Y	Pixel
1	19.89	20	6	(20,6)
2	19.78	20	7	(20,7)
3	19.67	20	8	(20,8)
4	19.56	20	9	(20,9)
5	19.45	19	10	(19,10)
6	19.34	19	11	(19,11)
7	19.23	19	12	(19,12)
8	19.12	19	13	(19,13)
9	19.01	19	14	(19,14)
10	18.9	19	15	(19,15)

b. (-5, 50), (-15,0)

Steps	X	Round(x)	Y	Pixel
0	-5	-5	50	(-5,50)
1	-5.2	-5	49	(-5.2,49)
2	-5.4	-5	48	(-5,48)
3	-5.6	-6	47	(-6,47)
4	-5.8	-6	46	(-6,46)
5	-6	-6	45	(-6,45)
6	-6.2	-6	44	(-6,44)
7	-6.4	-6	43	(-6,43)
8	-6.6	-7	42	(-7,42)
9	-6.8	-7	41	(-7,41)
10	-7	-7	40	(-7,40)

c. (-10, -10), (48, 24)

Steps	X	Y	Round(y)	Pixel
0	-10	-10	-10	(-10,-10)
1	-9	-9.41	-9	(-9,-9)
2	-8	-8.82	-9	(-8,-9)
3	-7	-8.23	-8	(-7,-8)
4	-6	-7.64	-8	(-6,-8)
5	-5	-7.05	-7	(-5,-7)
6	-4	-6.46	-6	(-4,-6)
7	-3	-5.87	-6	(-3,-6)
8	-2	-5.28	-5	(-2,-5)
9	-1	-4.69	-5	(-1,-5)
10	0	-4.1	-4	(0,-4)

D. (48, 24), (-10, -10)

Steps	X	Y	Round(y)	Pixel
0	48	24	24	(48,24)
1	47	23.41	23	(47,23)
2	46	22.82	23	(46,23)
3	45	22.23	22	(45,22)
4	44	21.64	22	(44,22)
5	43	21.05	21	(43,21)
6	42	20.46	20	(42,20)
7	41	19.87	20	(41,20)
8	40	19.28	19	(40,19)
9	39	18.69	19	(39,19)
10	38	18.1	18	(38,18)

E. (-1011, -2022), (-2022, -1011)

Steps	X	Round(x)	Y	Pixel
0	-1011	-1011	-2022	(-1011,-2022)
1	-1012	-1012	-2021	(-1012,-2021)
2	-1013	-1013	-2020	(-1013,-2020)
3	-1014	-1014	-2019	(-1014,-2019)
4	-1015	-1015	-2018	(-1015,-2018)
5	-1016	-1016	-2017	(-1016,-2017)
6	-1017	-1017	-2016	(-1017,-2016)
7	-1018	-1018	-2015	(-1018,-2015)
8	-1019	-1019	-2014	(-1019,-2014)
9	-1020	-1020	-2013	(-1020,-2013)
10	-1021	-1021	-2012	(-1021,-2012)

pixel count: max{(x1-x0), (y1-y0)} +1

Q. Starting point (2,2) and ending point (x,y)
m= 0.6, x increases 5 times.
Determine the ending point

ans. x1 = x0 + 5 = 2+5 =7
m= y1-y0/x1-x0
0.6 = y1- 2/ 7-2
y1 -2 = 06*5
y1= 3+2 =5 //(x1,y1) = (7,5)

number pixel = max {(7-2), (5-2)} +1 = 6 pixels