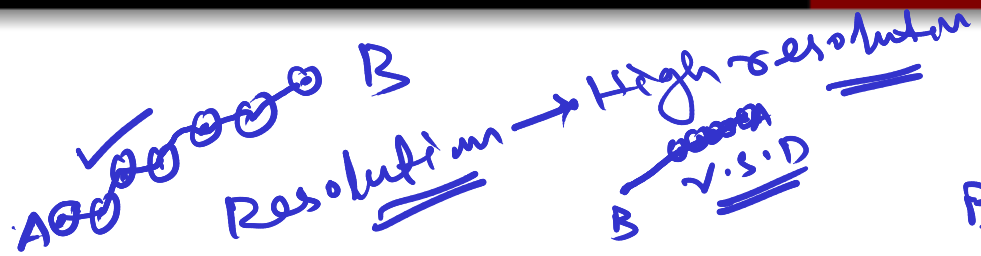
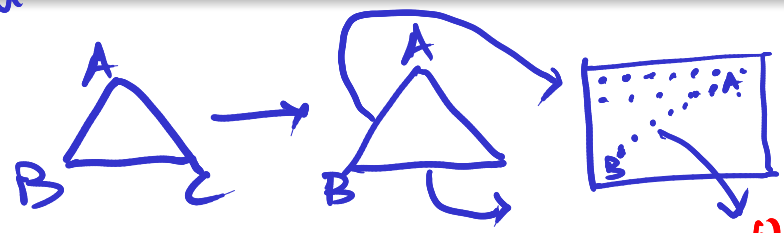




COM524T-Interactive Computer Graphics

Line Drawing Algorithms

output primitives
① Point → glVertex (x,y)
② Line → (x1,y1) (x2,y2) (x3,y3) ...
point
(x0,y0) (x1,y1)
y = mx + c
... → glVertex

Dr. Ram Prasad Padhy

Computer Science & Engineering, IIITDM Kancheepuram

Table of contents

What is Digital Differentiation?

- Differentiation of a continuous function $f(x)$ with respect to x:

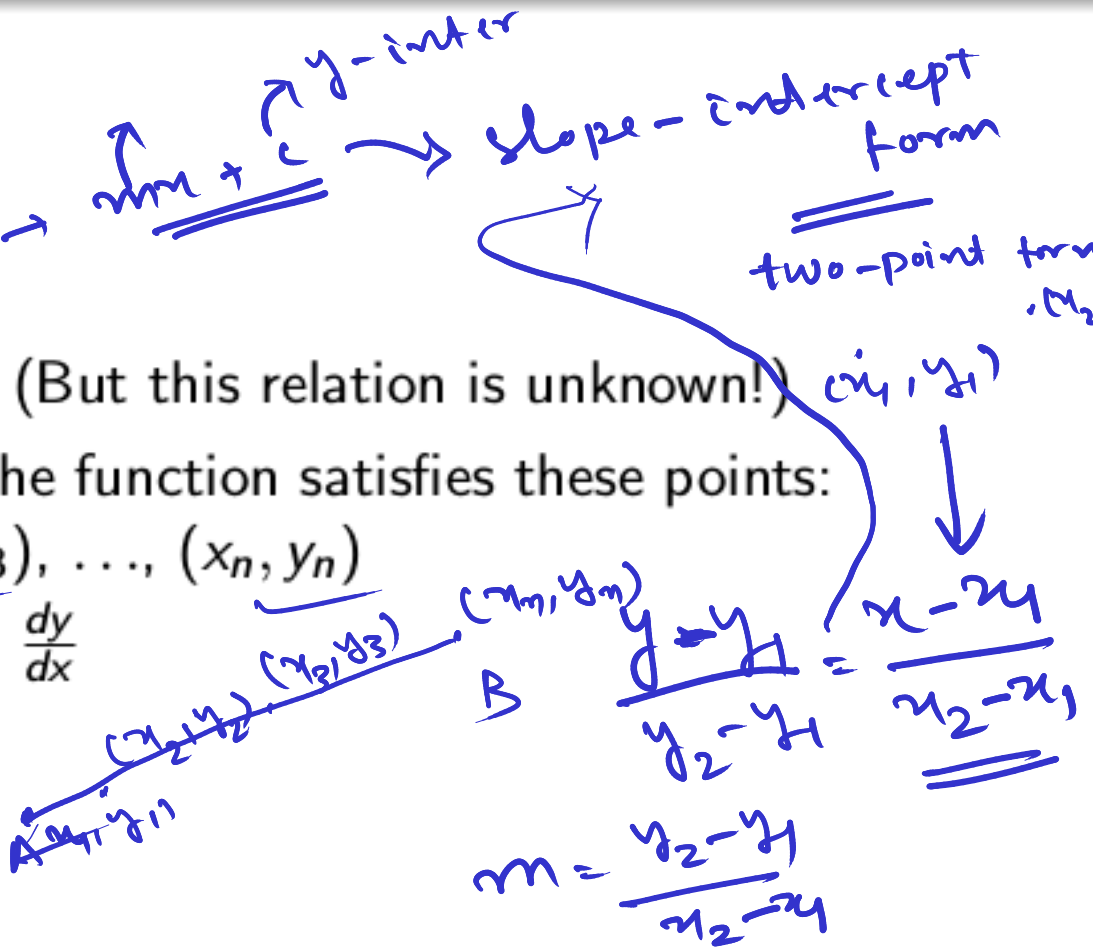
$$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{(x+h) - x}$$

- h is the minimum change in x (Δx) which causes a change in $f(x) = y =$
- For discrete plane: $h \neq 1$
- Differentiation of a discrete function $f(x)$ with respect to x:

$$\frac{df}{dx} = \frac{f(x+1) - f(x)}{(x+1) - x} = (f(x+1) - f(x))$$

Digital Differentiation on a function

- Let $y = f(x) = ax + b$ (But this relation is unknown!)
- In raster (x, y) plane, the function satisfies these points: (x_1, y_1), (x_2, y_2), (x_3, y_3), ..., (x_n, y_n)
- Our objective is to find $\frac{dy}{dx}$



Digital Differentiation on a function

- Let $\frac{dy}{dx} = f'(x)$
- $f'(x)$ can be defined as:
  $f'(x) = (f(x+1) - f(x))$
- $f'(x)|_{x=x_i}$
- $= (f(x+1)|_{x=x_{i+1}} - f(x)|_{x=x_i})$
- $= (y_{i+1} - y_i)$

$f' = \frac{y_2 - y_1}{x_2 - x_1}$ → (1)

→ discrete fun^n

$(x+1) - x = 1$

$f' = y_2 - y_1$

$= y_{i+1} - y_i$

$f(x)|_{x=x_{i+1}}$

$y_{i+1} = f' + y_i$

(x_i, y_i)

(x_{i+1}, y_{i+1})

$f' = y_{i+1} - y_i$

$f'(x)|_{x=x_i}$

(x_i, y_i)

DDA Line Drawing Algorithm
(DDA: Digital Differential Analyzer)

Formation of DDA

- Cartesian *slope-intercept equation* for a straight line:
  $y = m.x + b$
- $m = \frac{y_2 - y_1}{x_2 - x_1}$
- $b = y_1 - m \cdot x_1$
  OR
  $b = y_2 - m \cdot x_2$

(x_2, y_2)

(x_1, y_1)

$y_1 = mx_1 + b$

$\Rightarrow b = y_1 - mx_1$

$y_2 = mx_2 + b$

$\Rightarrow b = y_2 - mx_2$

Formation of DDA

- For any given interval Δx along a line,
  Δy can be found as:
  $\Delta y = m \Delta x$
- For any given interval Δy along a line,
  Δx can be found as:
  $\Delta x = \frac{\Delta y}{m}$

Formation of DDA, when $|m| \leq 1$

- When $|m| \leq 1$,
  with a given point (x_k, y_k) lying on the line,
  the next point can be found as:
  $y_{k+1} - y_k = m(x_{k+1} - x_k)$
  where, $\Delta y = (y_{k+1} - y_k)$, and $\Delta x = (x_{k+1} - x_k)$

Formation of DDA, when $|m| \leq 1$

- We vary values of x incrementally,
  i.e., $x_{k+1} = x_k + 1$
- Hence the equation $y_{k+1} - y_k = m(x_{k+1} - x_k)$ reduces to,
  $y_{k+1} = y_k + m$
- If we start from a point and vary values of x in decreasing order,
  i.e., $x_{k+1} = x_k - 1$
- Hence the equation $y_{k+1} - y_k = m(x_{k+1} - x_k)$ reduces to,
  $y_{k+1} = y_k - m$

Example of DDA algorithm when $|m| \leq 1$

Case - I: the algorithm starts from the point with less abscissa value

Example of DDA algorithm when $|m| \leq 1$

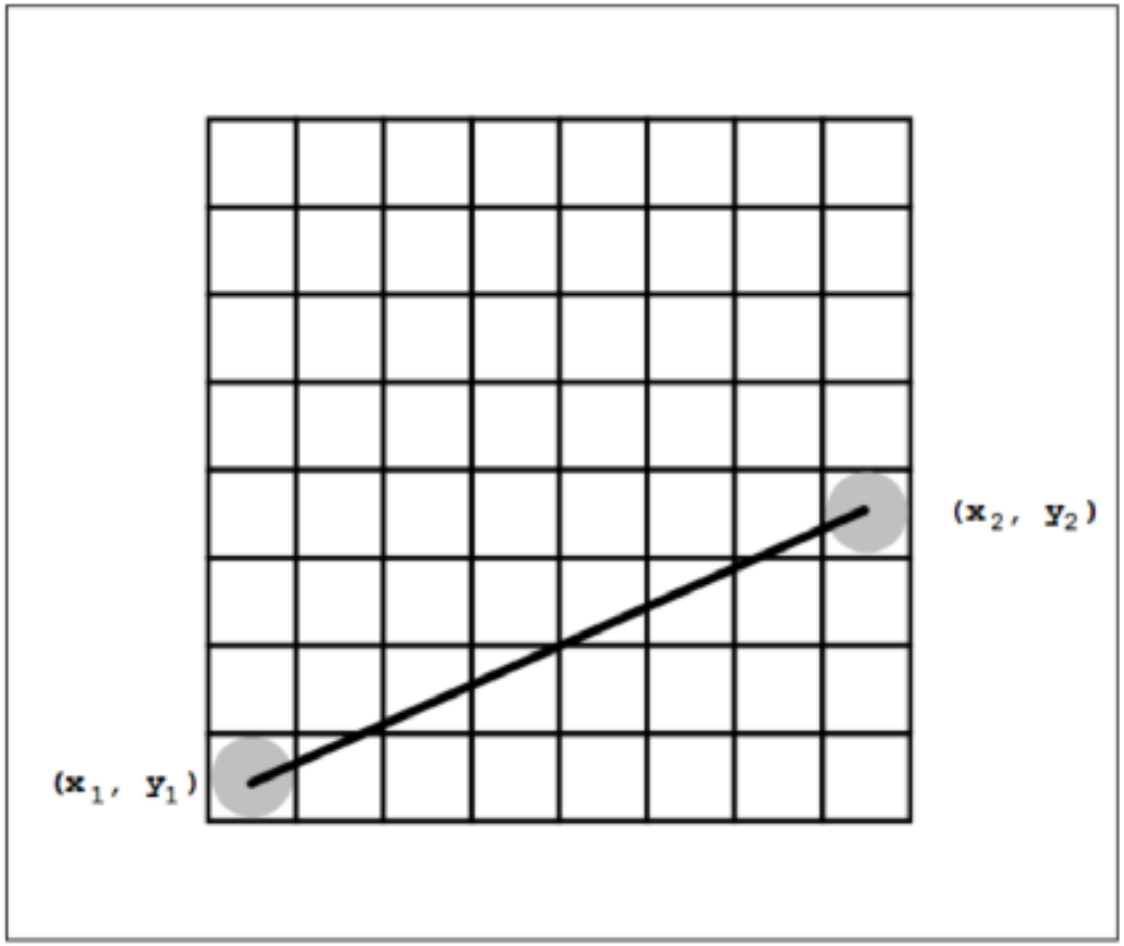


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

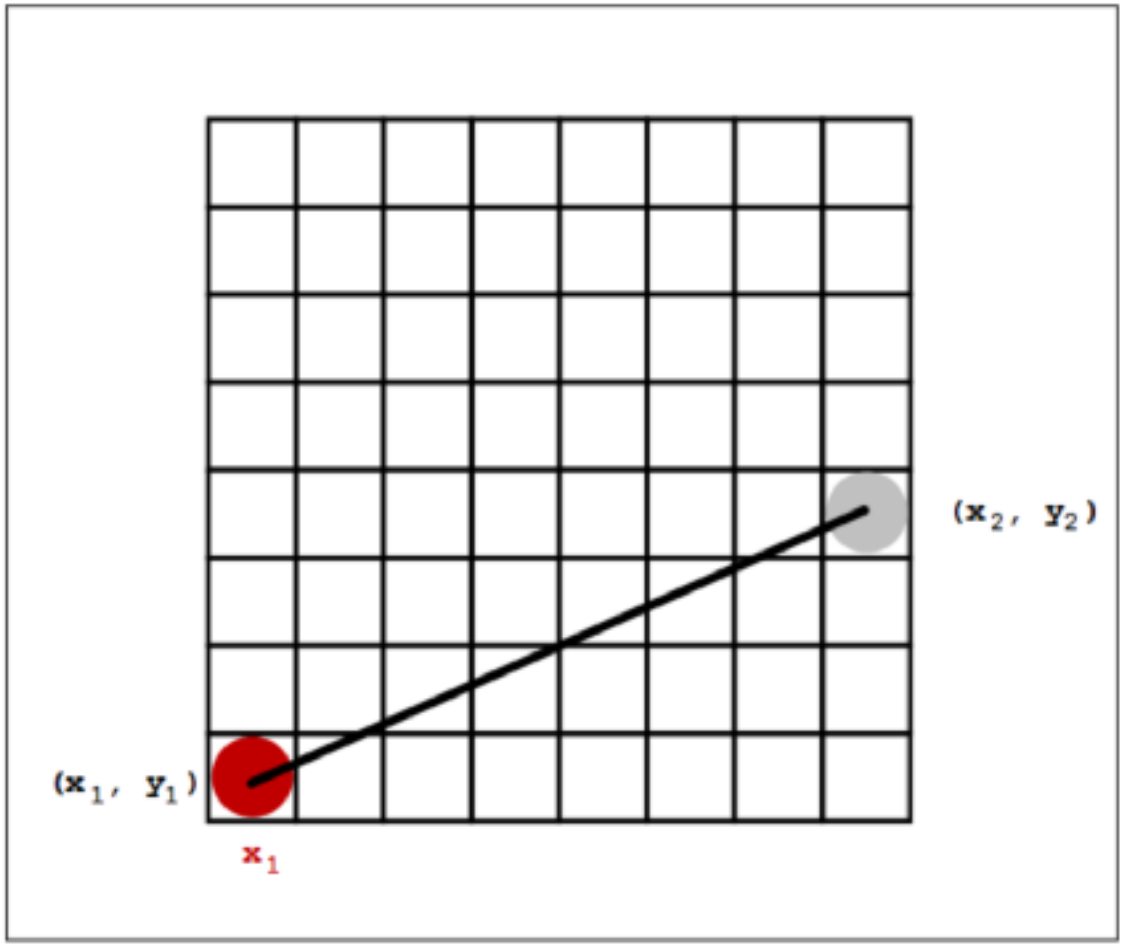


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

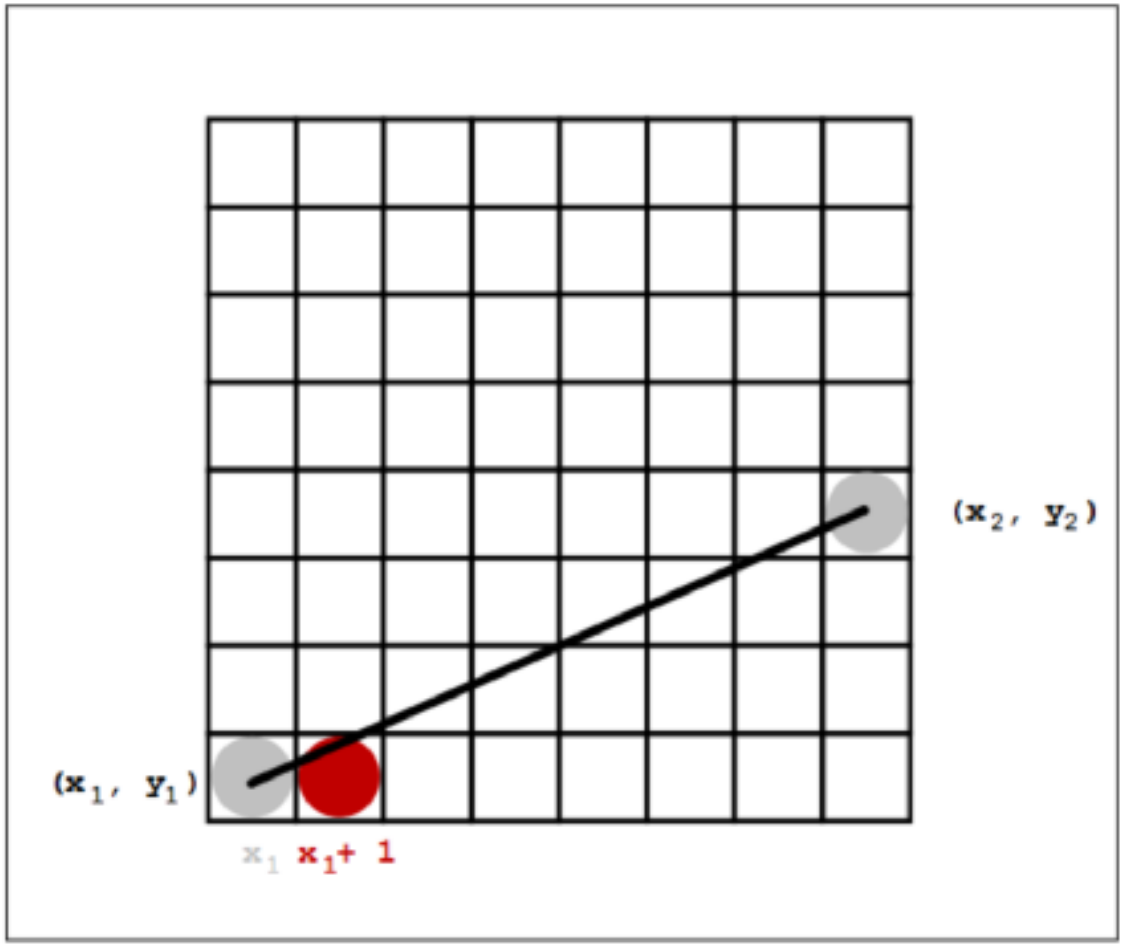


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

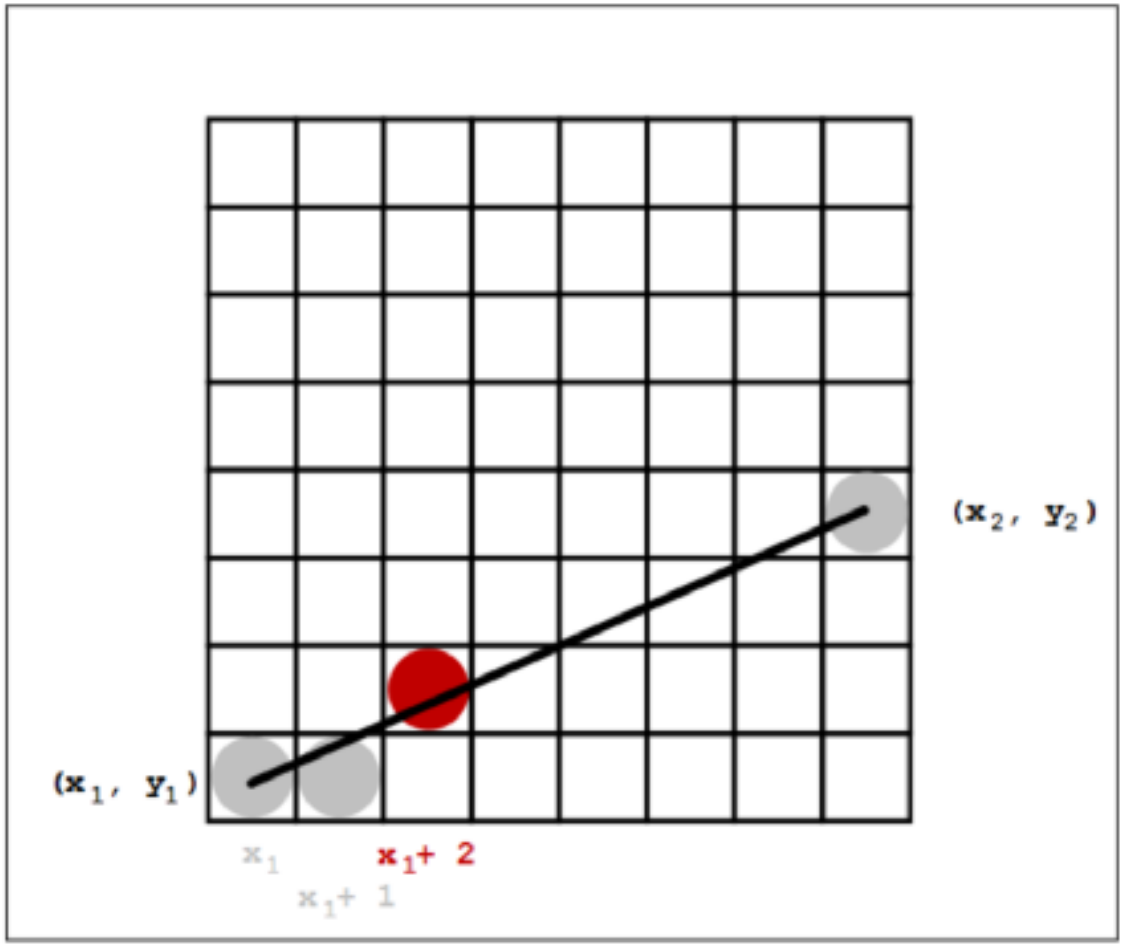


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

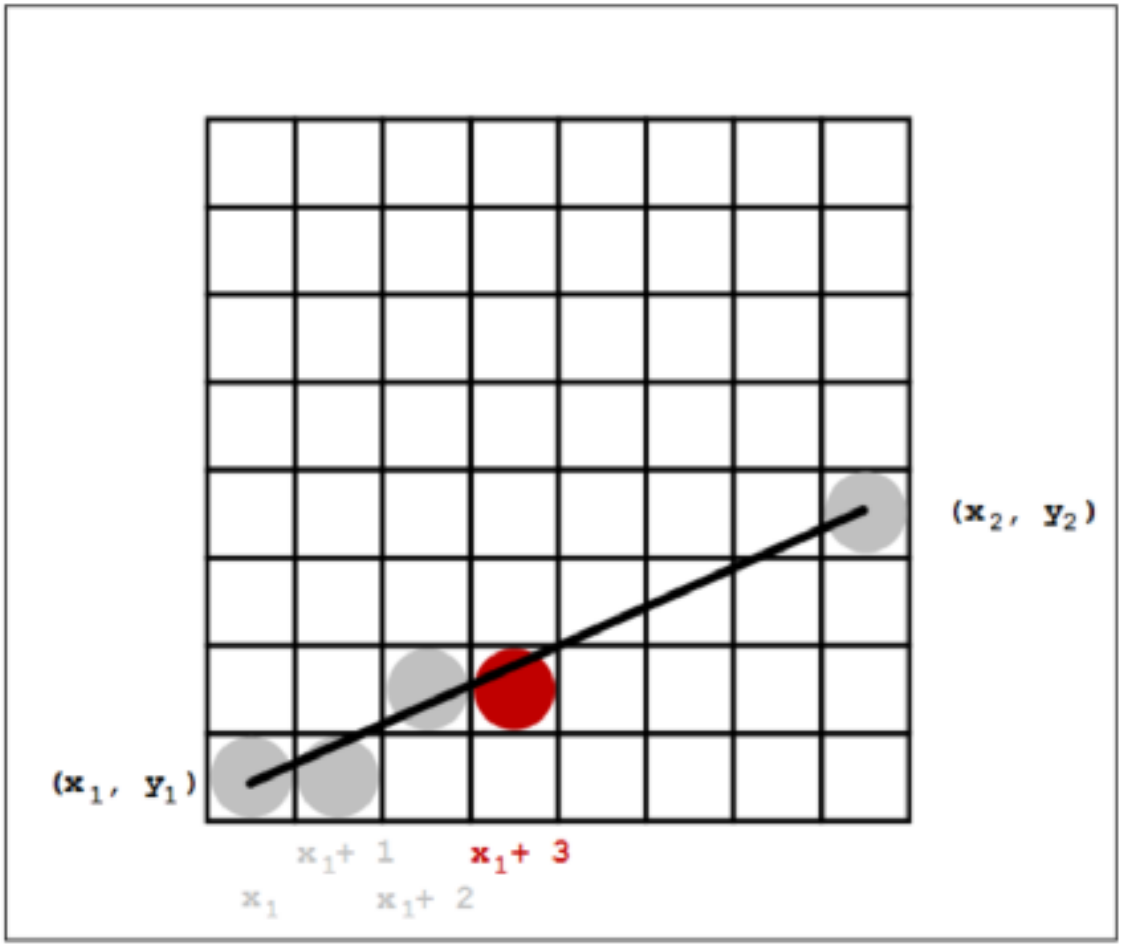


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

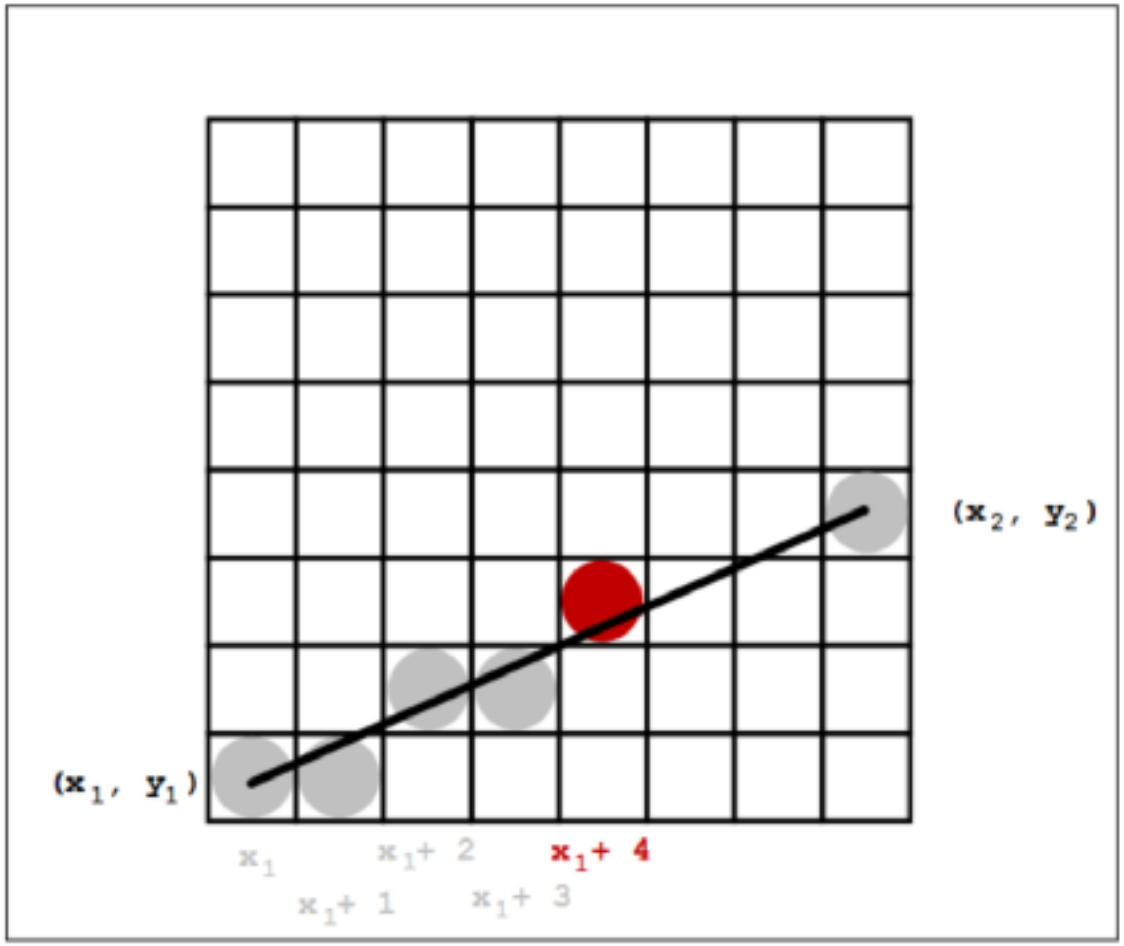


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

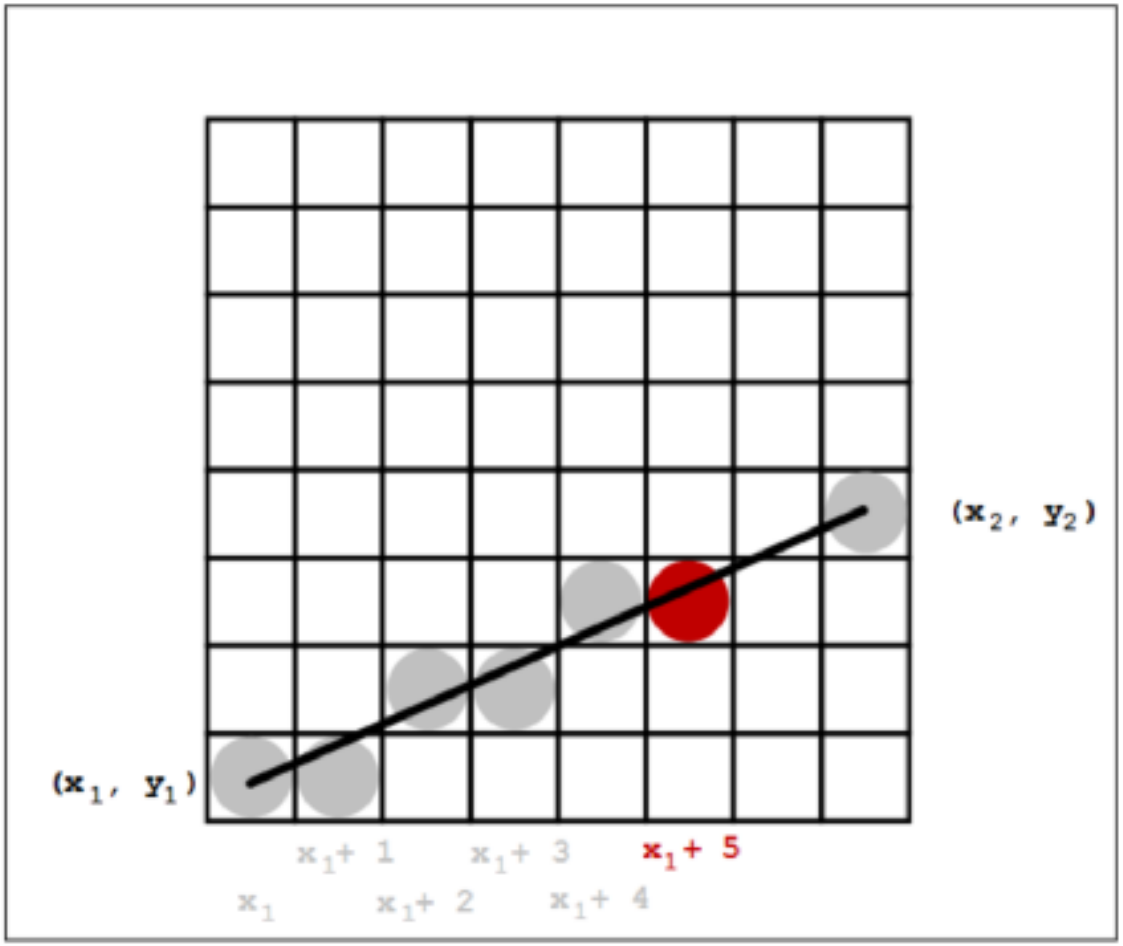


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

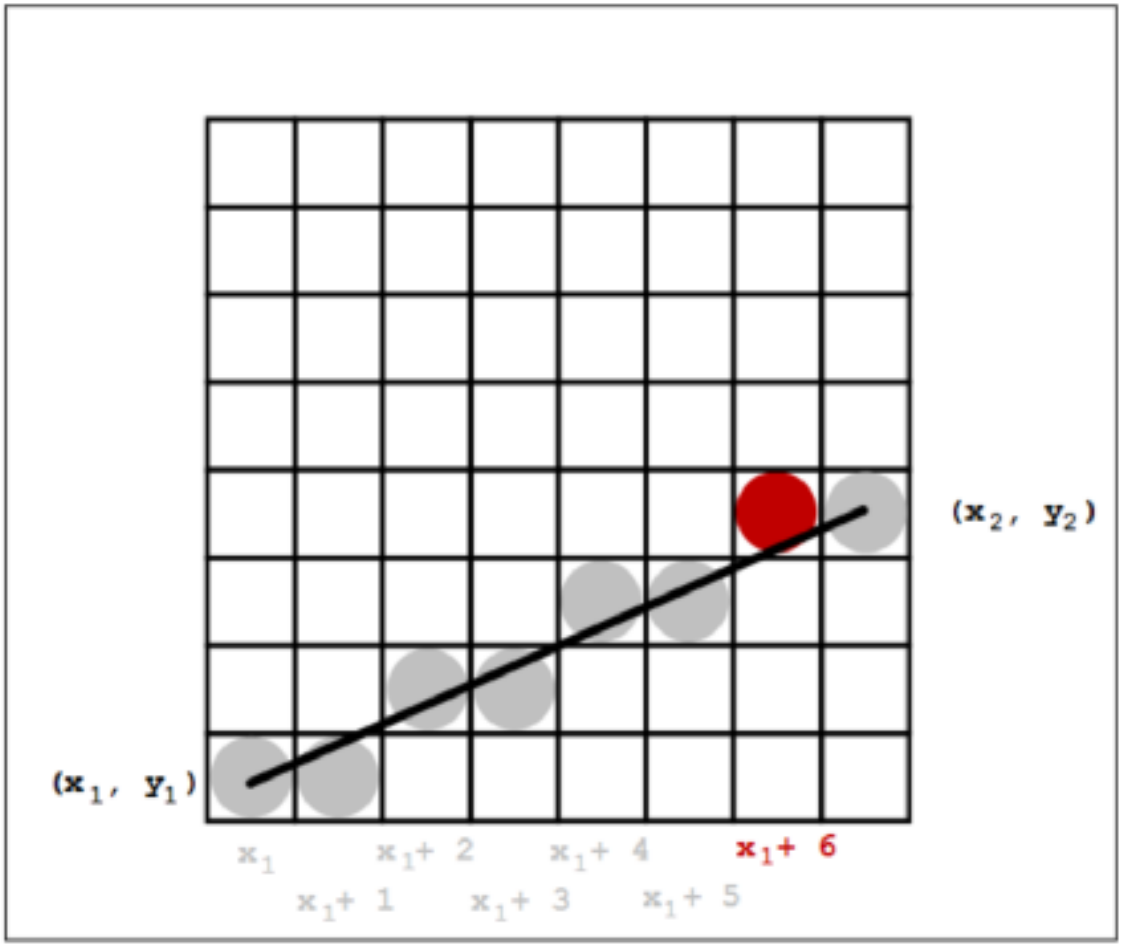


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

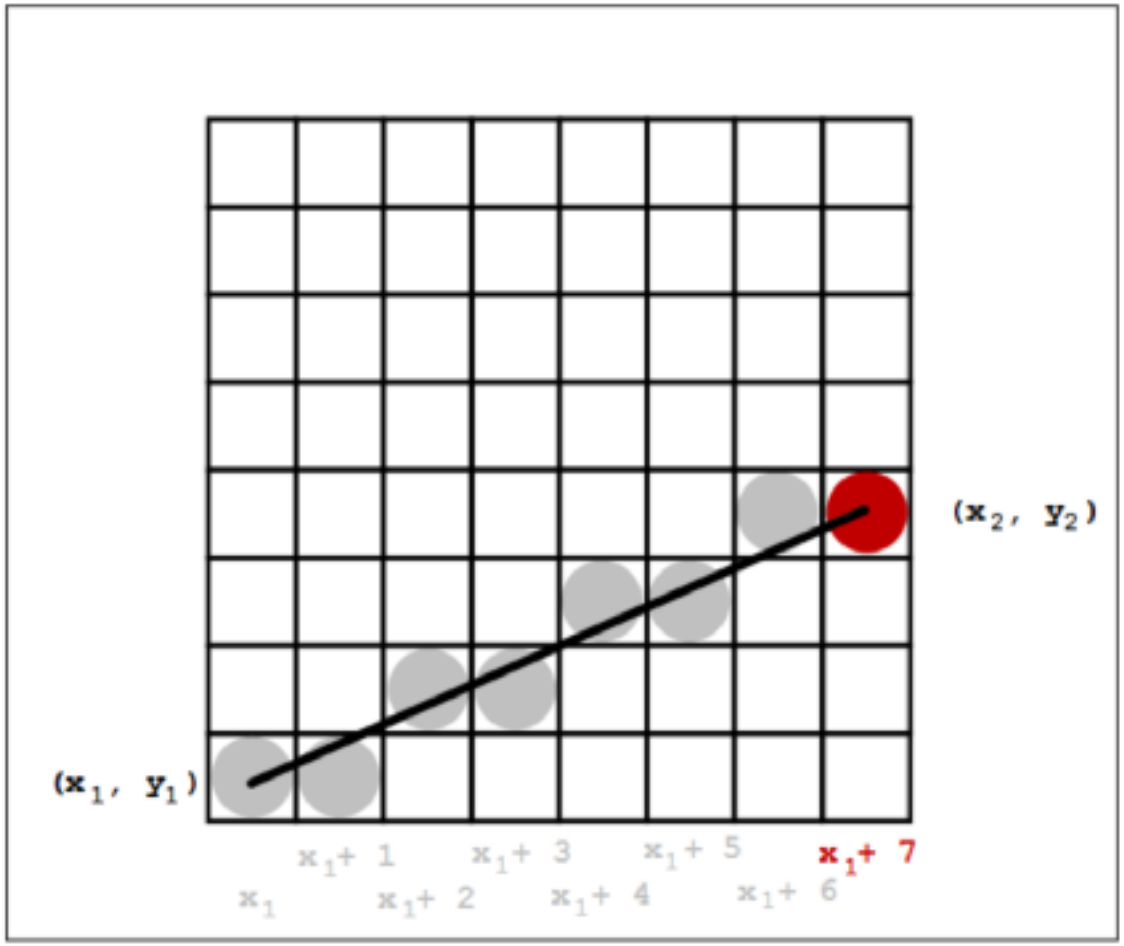


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

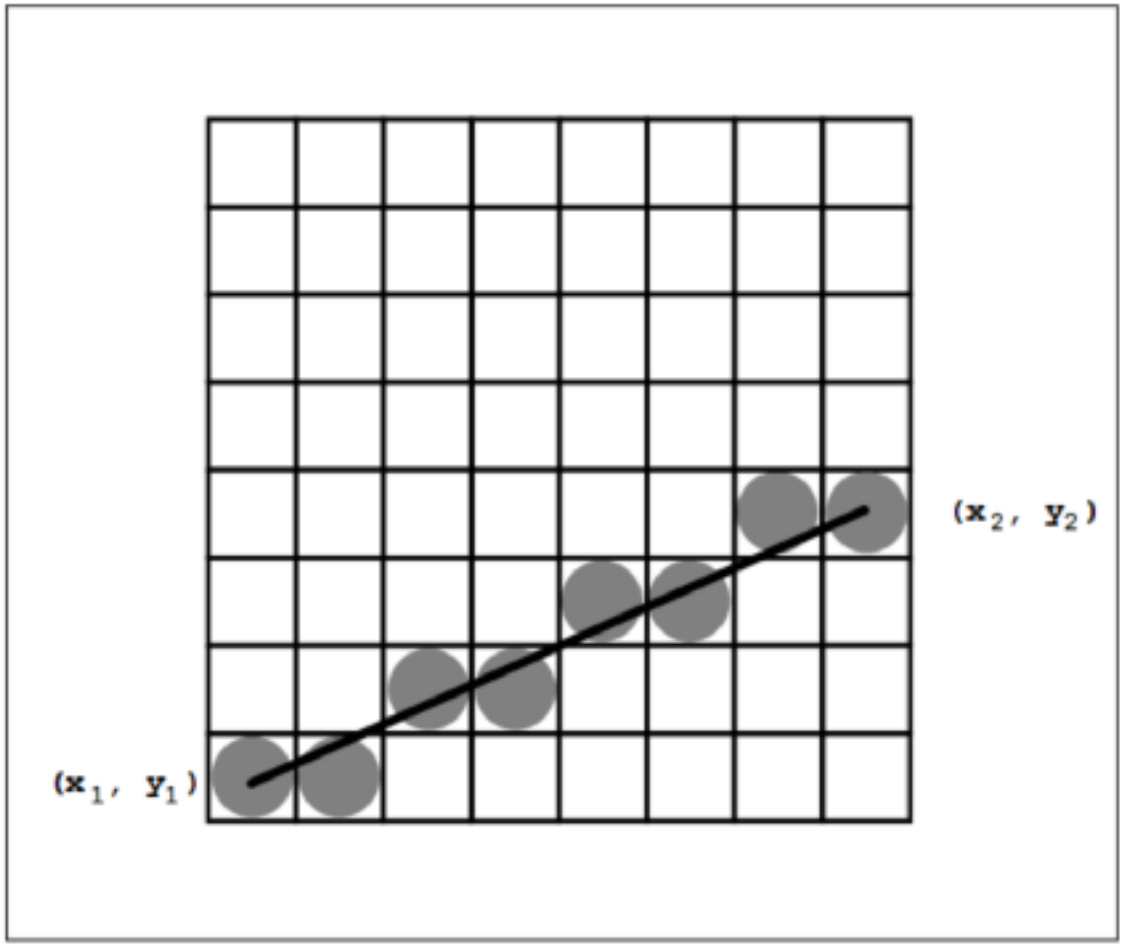


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

Case - II: the algorithm starts from the point with more abscissa value

Example of DDA algorithm when $|m| \leq 1$

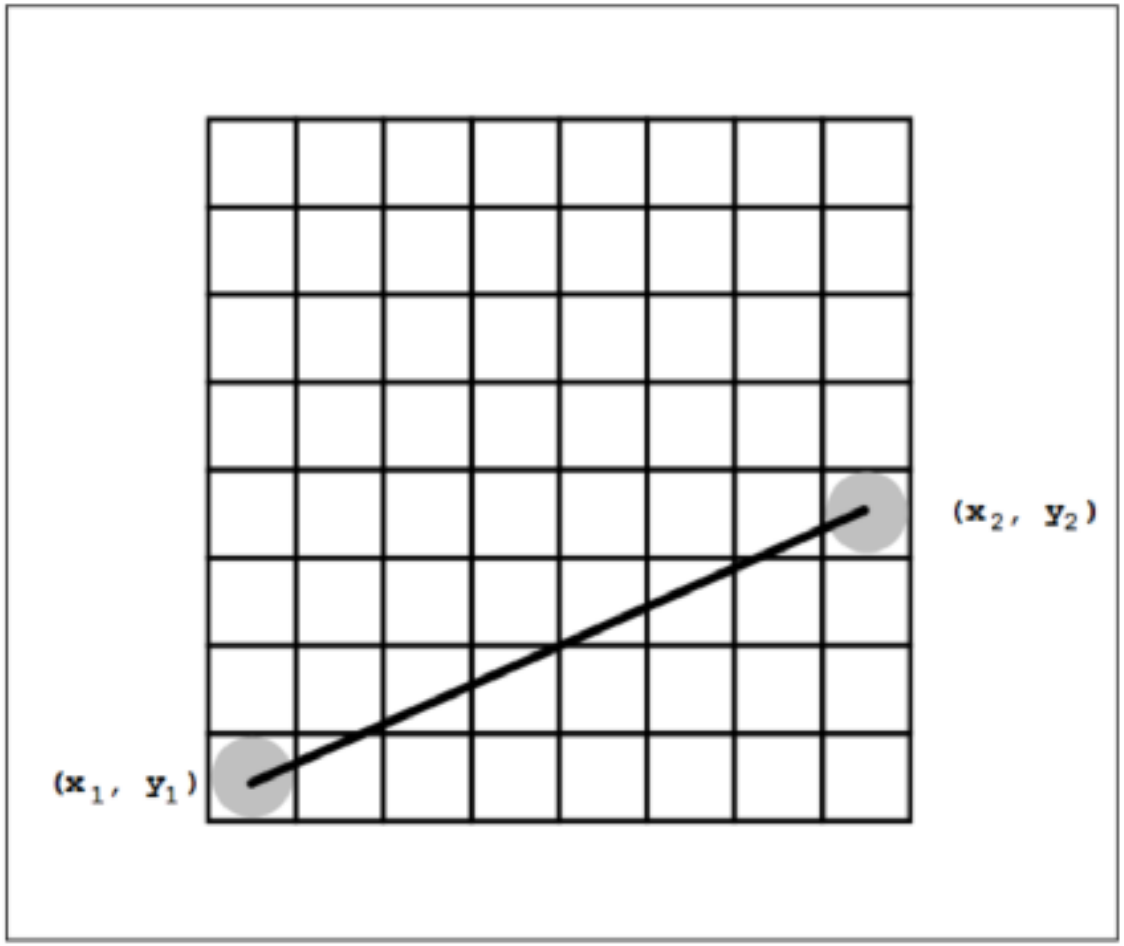


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

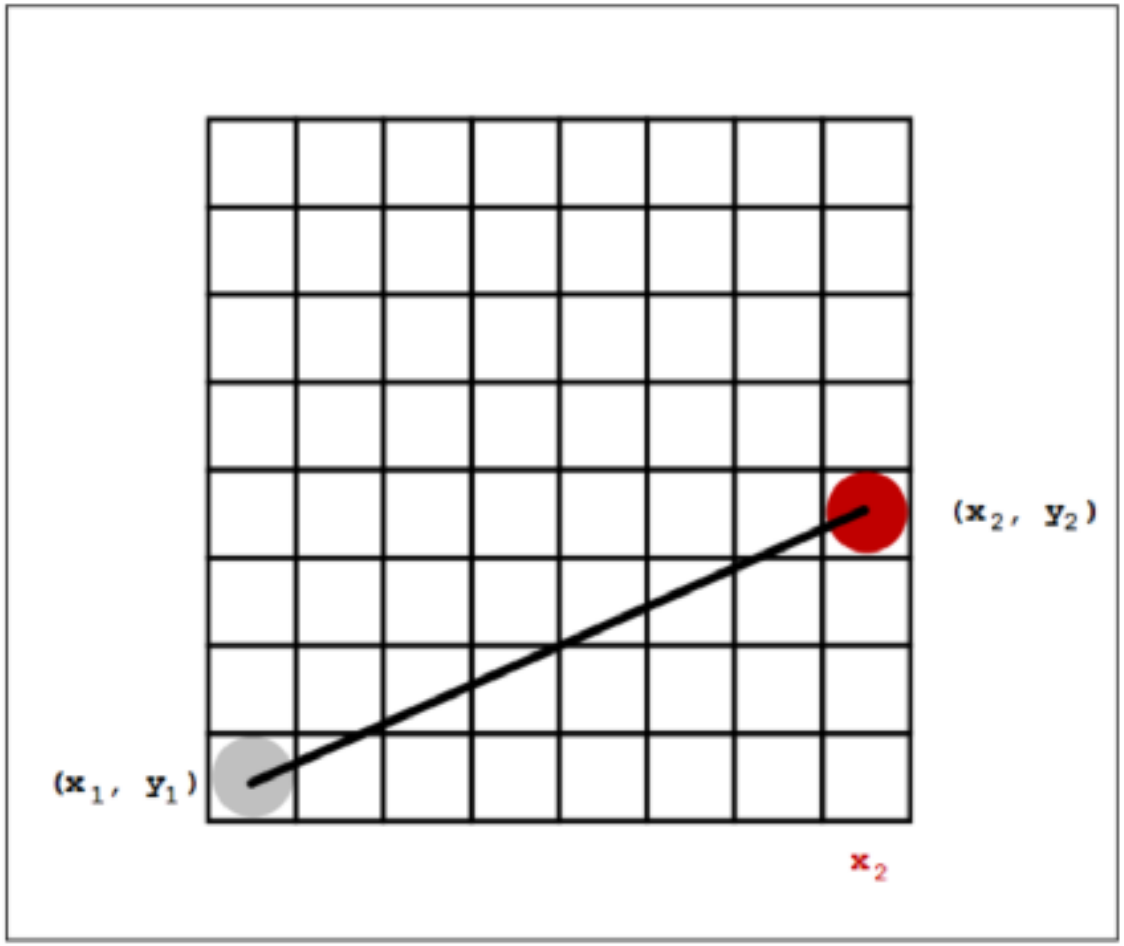


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

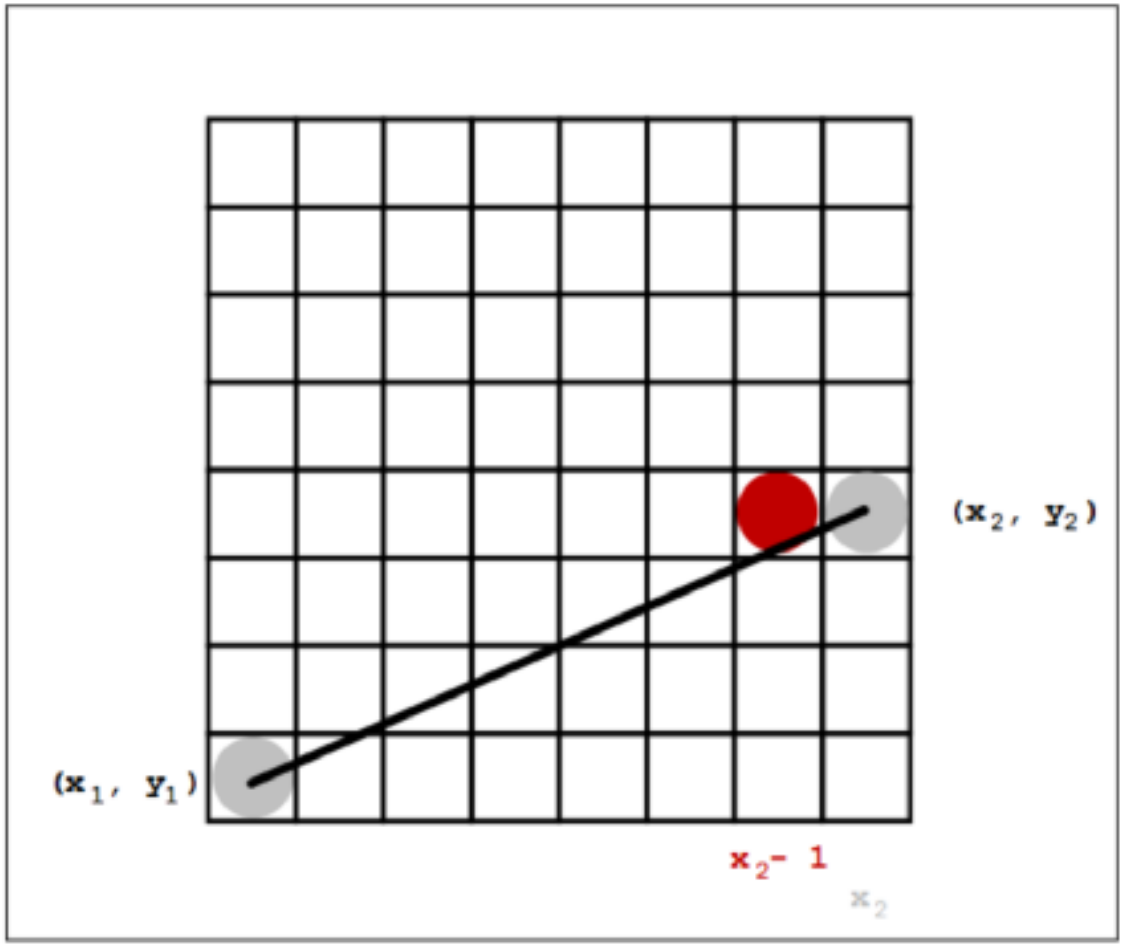


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

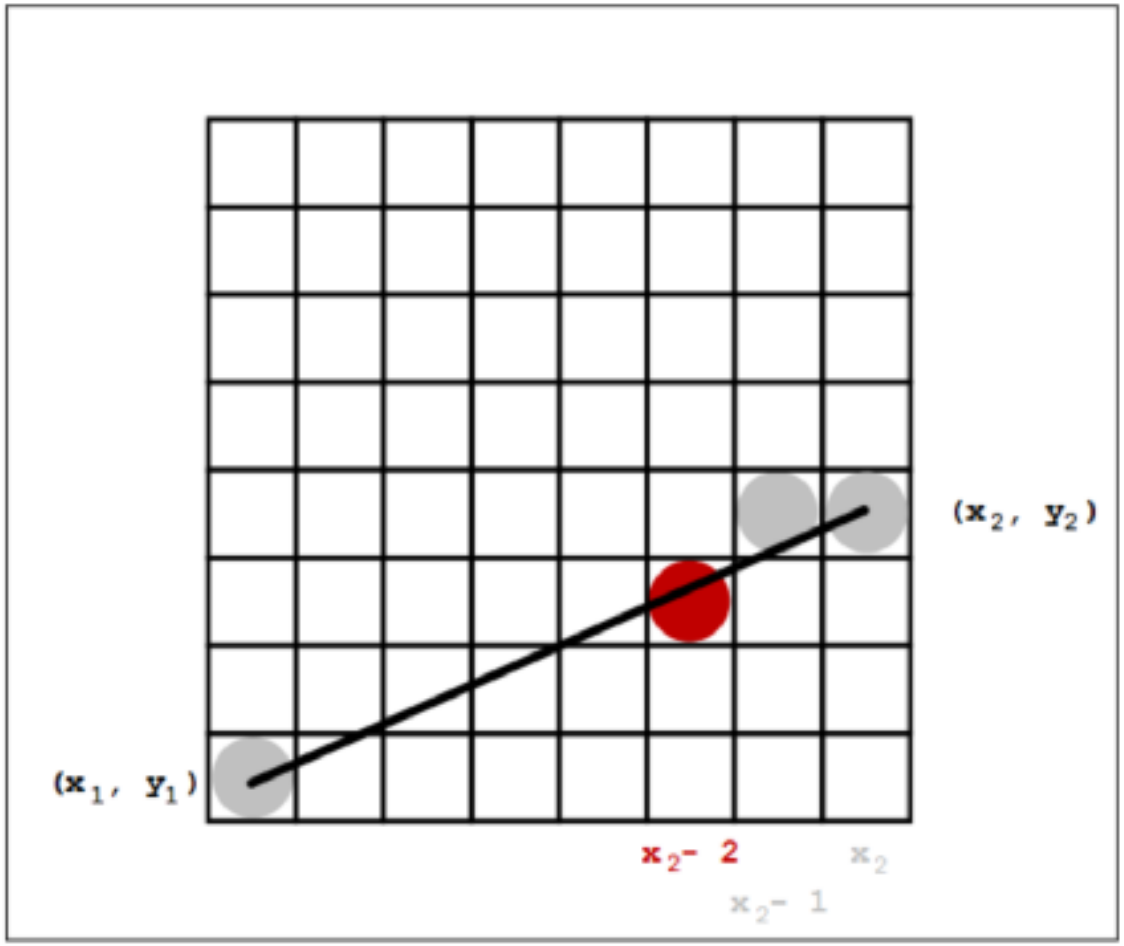


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

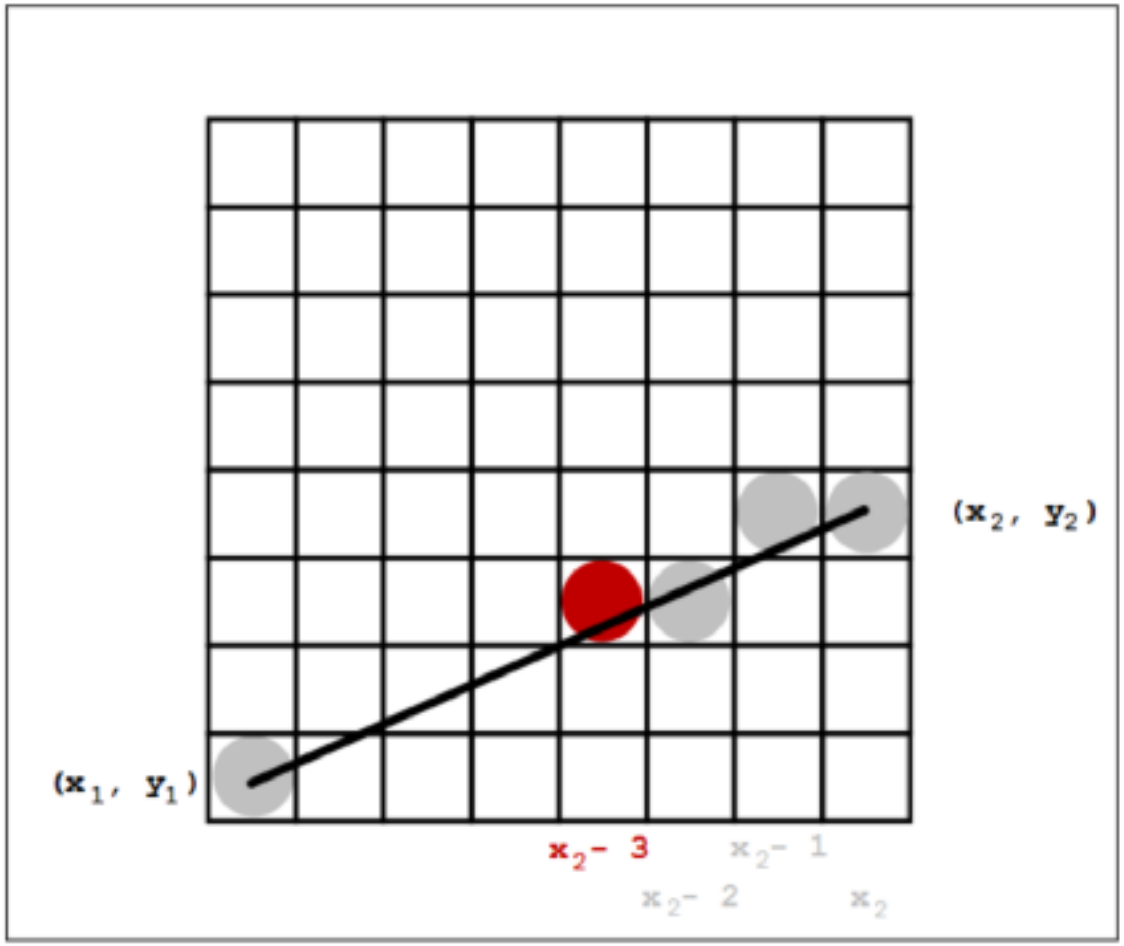


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

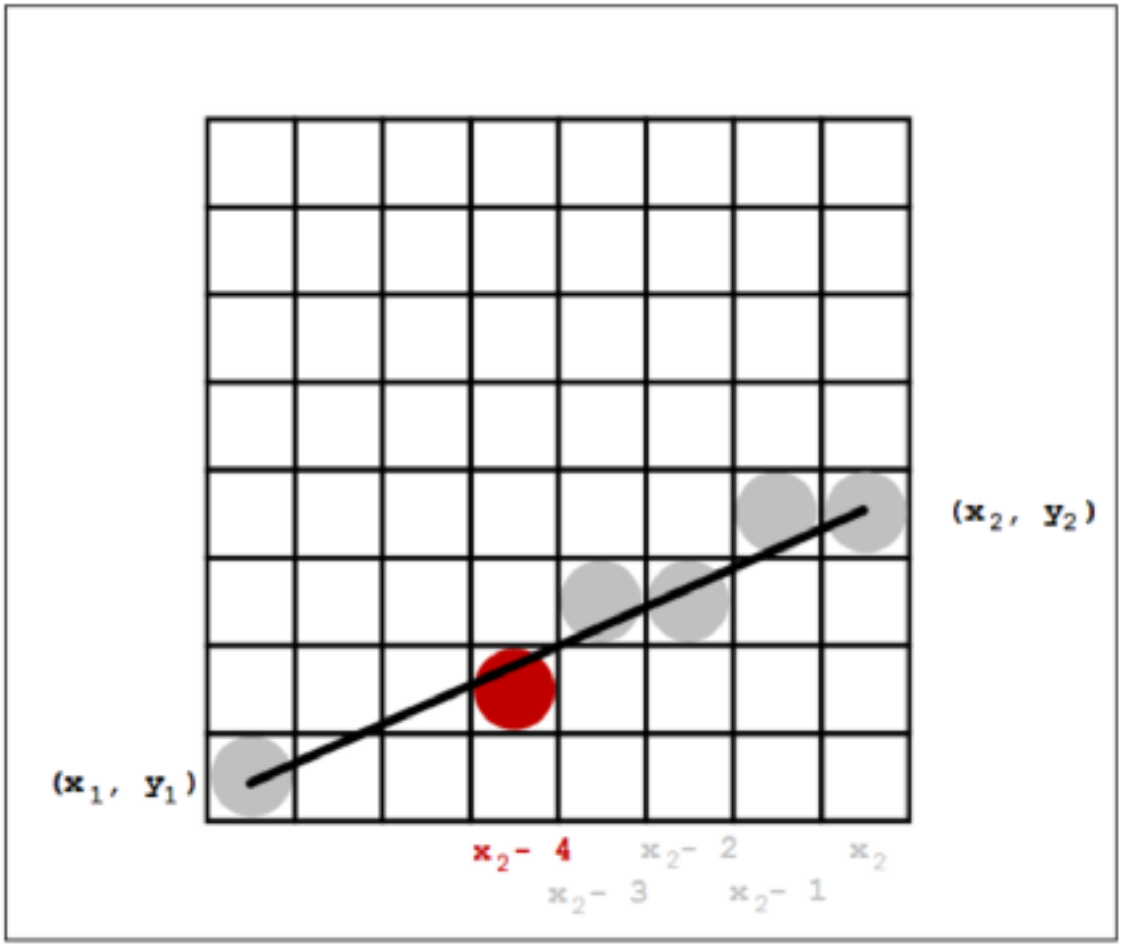


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

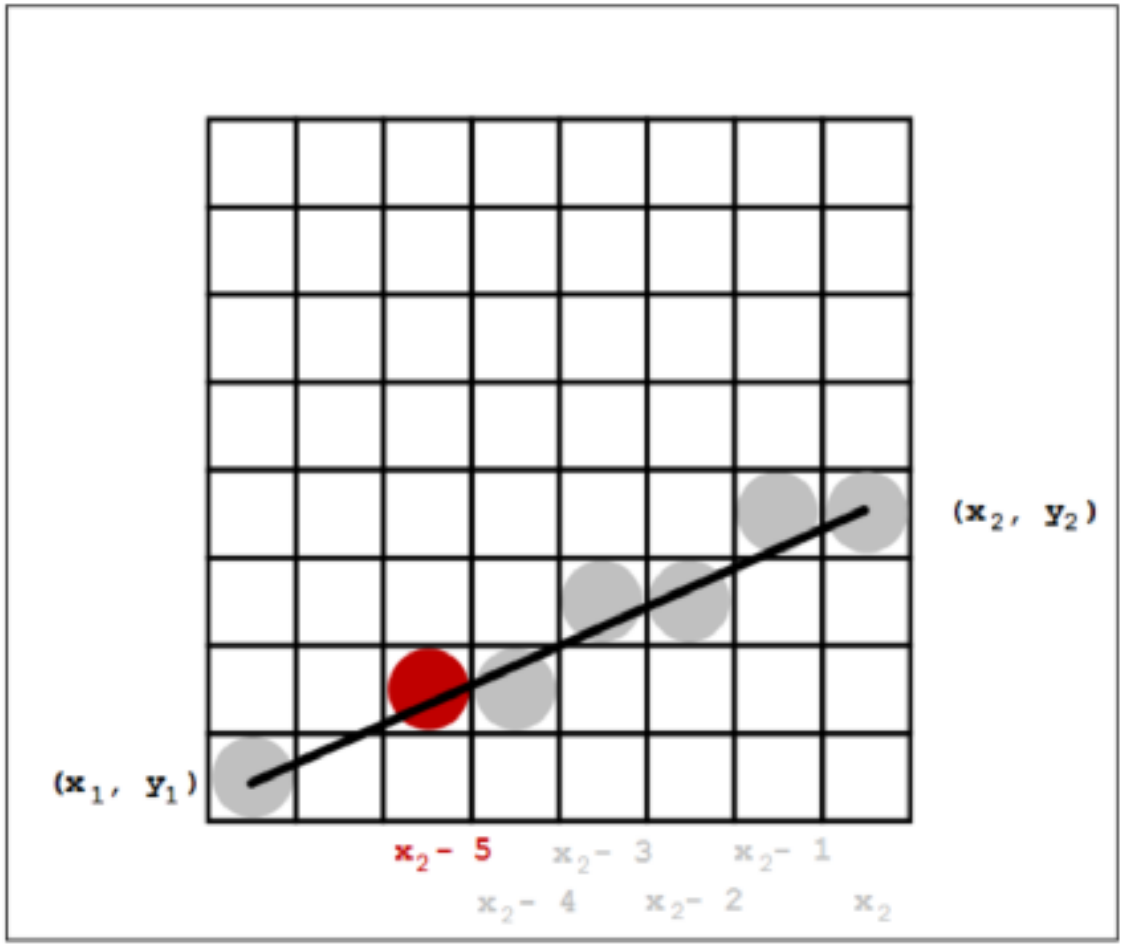


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

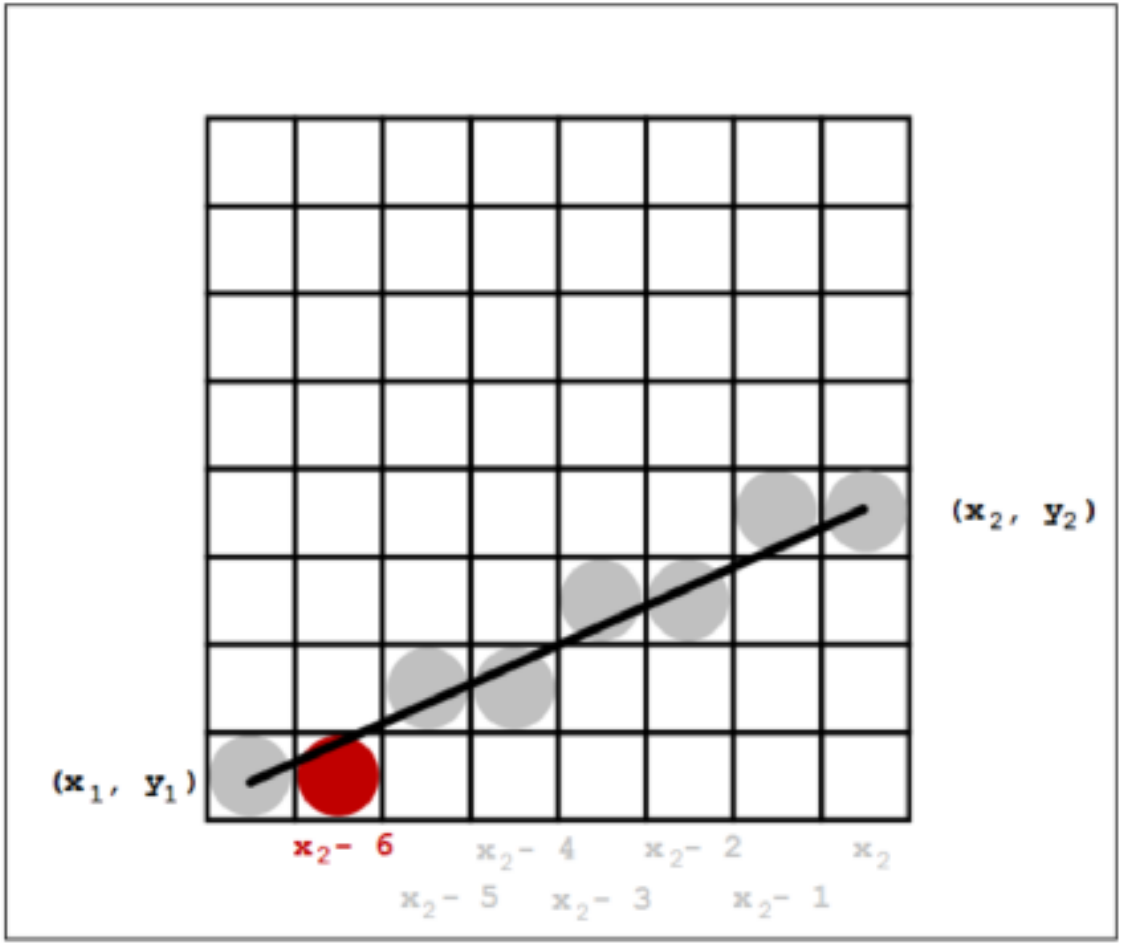


Figure : An example of DDA algorithm when $|m| \leq 1$

Example of DDA algorithm when $|m| \leq 1$

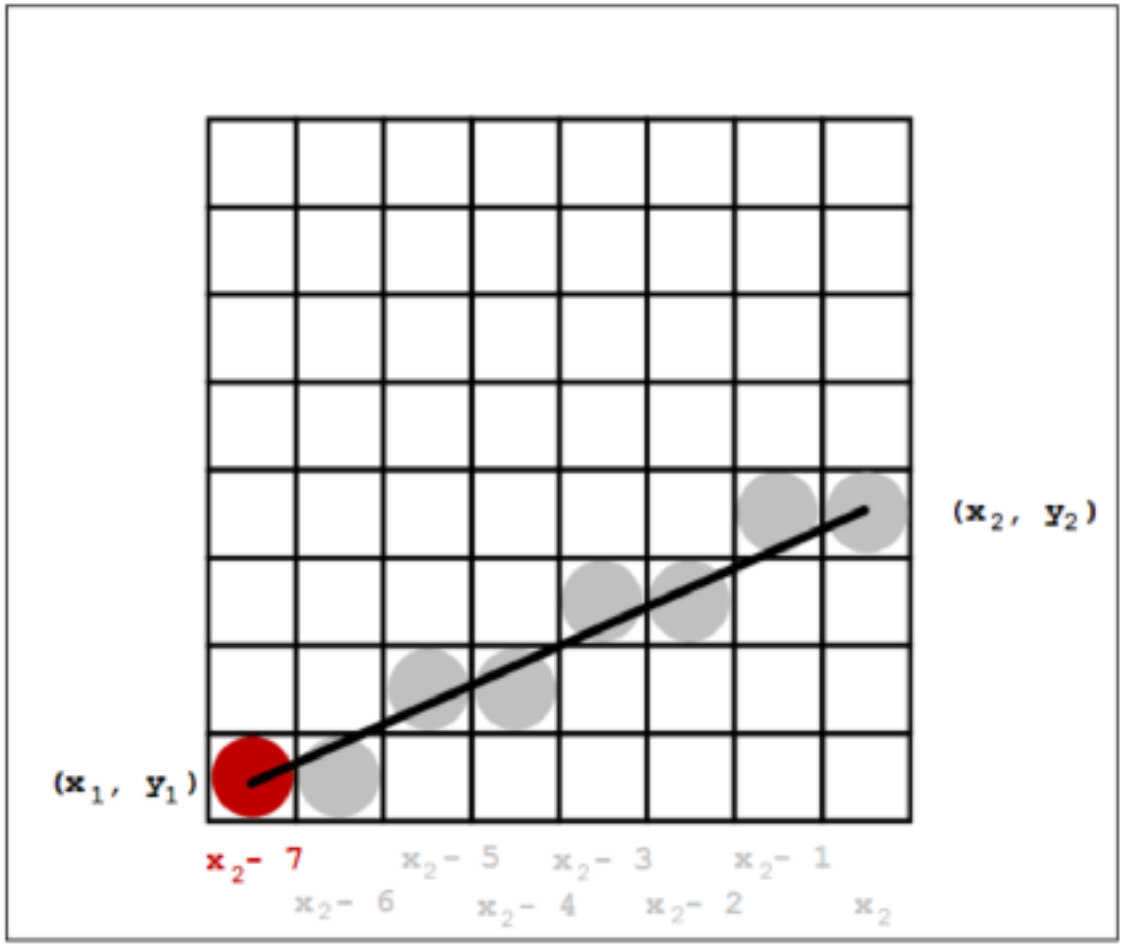


Figure : An example of DDA algorithm when $|m| < 1$

Example of DDA algorithm when $|m| \leq 1$

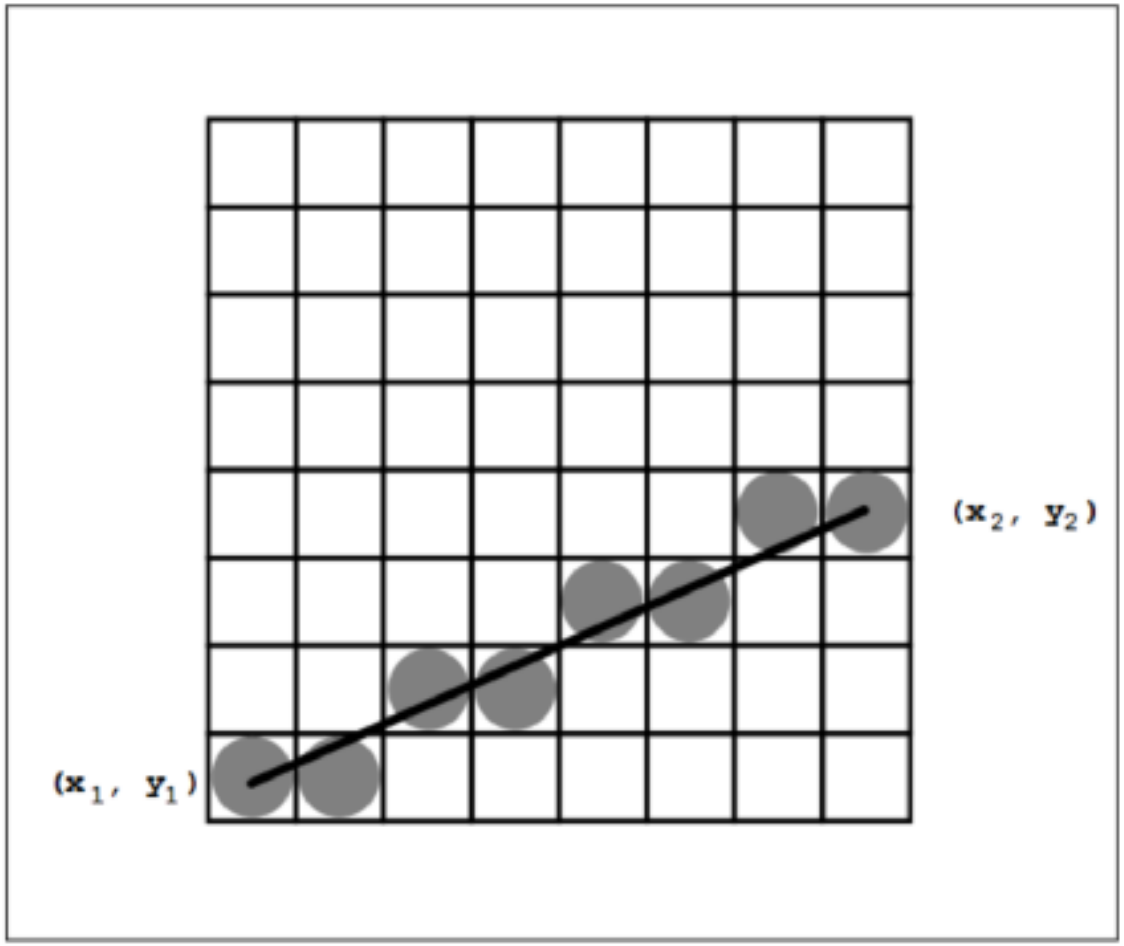


Figure : An example of DDA algorithm when $|m| \leq 1$

Formation of DD, when $|m| > 1$

- When $|m| > 1$,
  with a given point (x_k, y_k) lying on the line,
  the next point can be found as:
  $x_{k+1} - x_k = \frac{(y_{k+1} - y_k)}{m}$
  where, $\Delta x = (x_{k+1} - x_k)$, and $\Delta y = (y_{k+1} - y_k)$

Formation of DD, when $|m| > 1$

- We vary values of y incrementally,
  i.e., $y_{k+1} = y_k + 1$
- Hence the equation $x_{k+1} - x_k = \frac{(y_{k+1} - y_k)}{m}$ reduces to,
  $x_{k+1} = x_k + \frac{1}{m}$

- If we start from a point and vary values of y in decreasing order,
  i.e., $y_{k+1} = y_k - 1$
- Hence the equation $x_{k+1} - x_k = \frac{(y_{k+1} - y_k)}{m}$ reduces to,
  $x_{k+1} = x_k - \frac{1}{m}$

Example of DDA algorithm when $|m| > 1$

Case - I: the algorithm starts from the point with less ordinate value

Example of DDA algorithm when $|m| > 1$

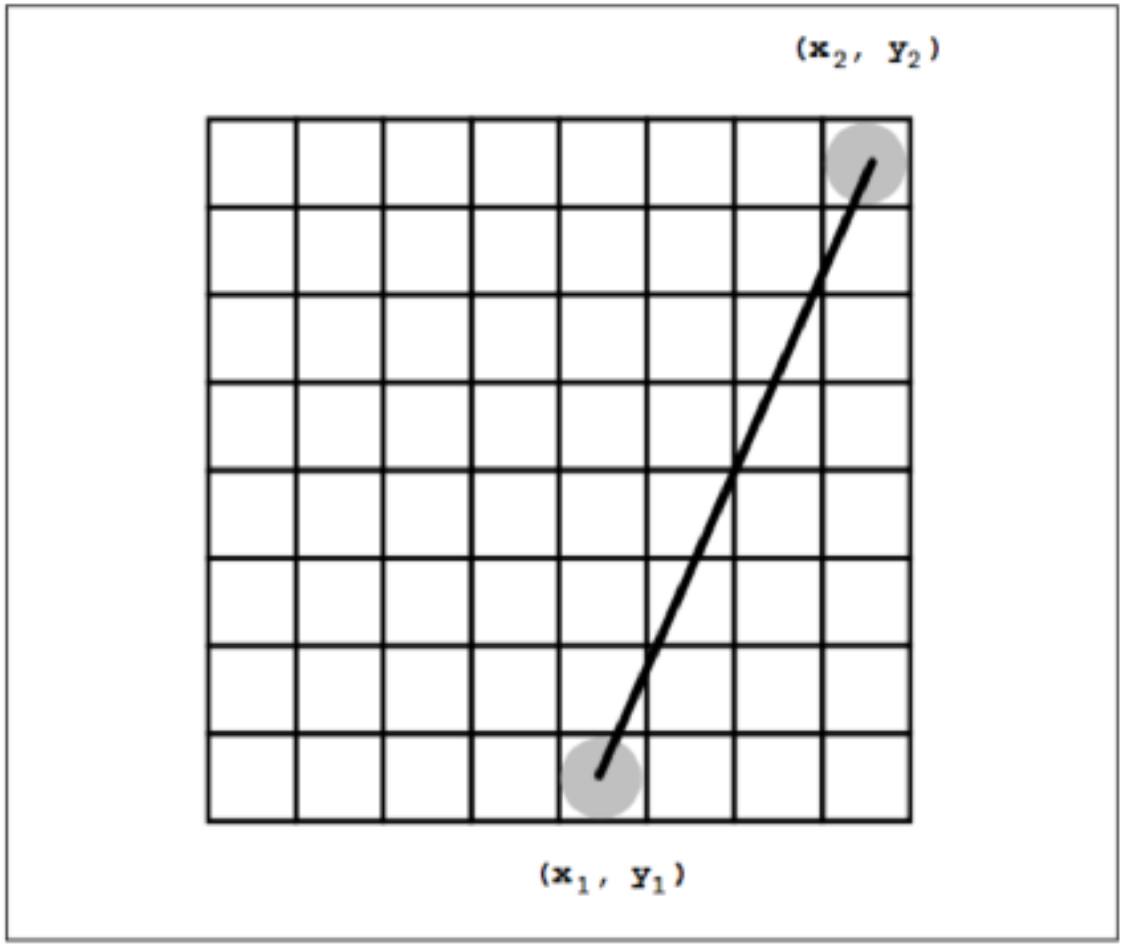


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

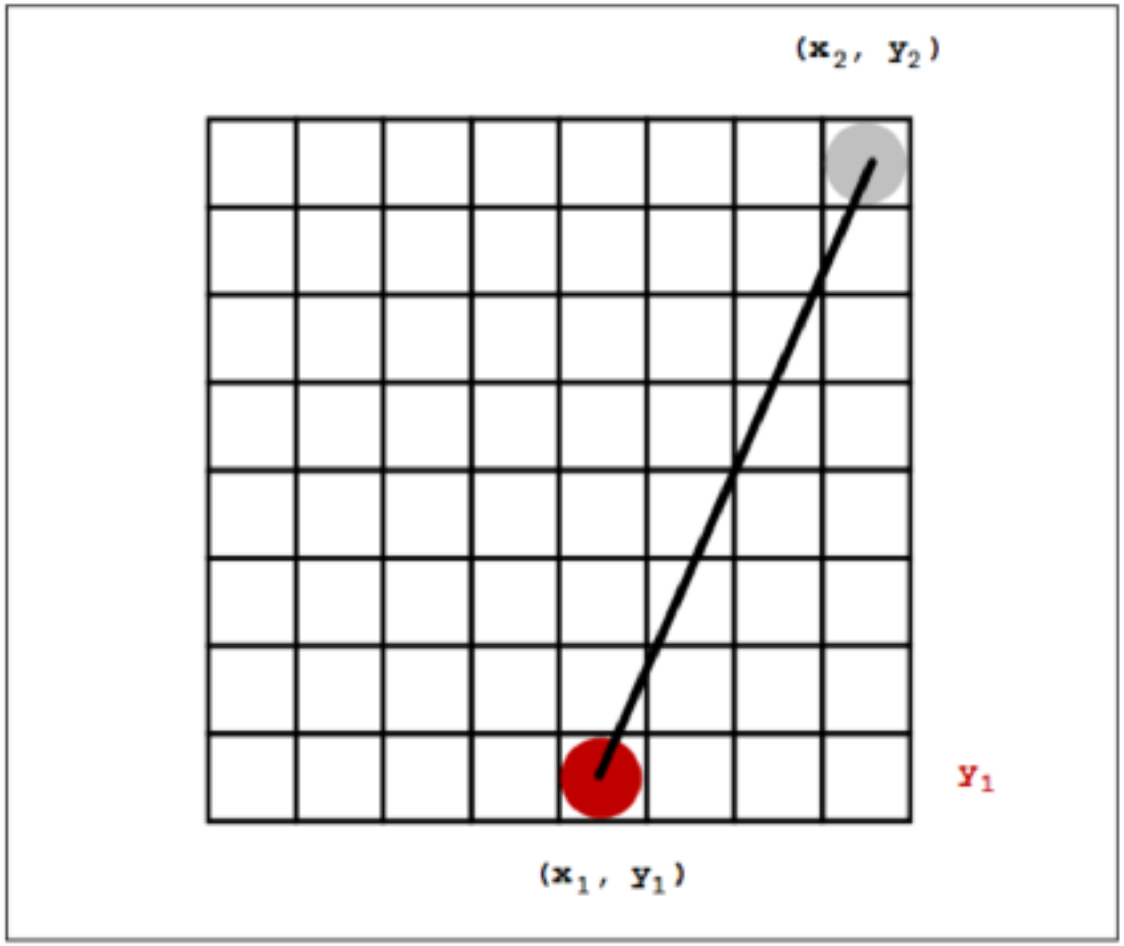


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

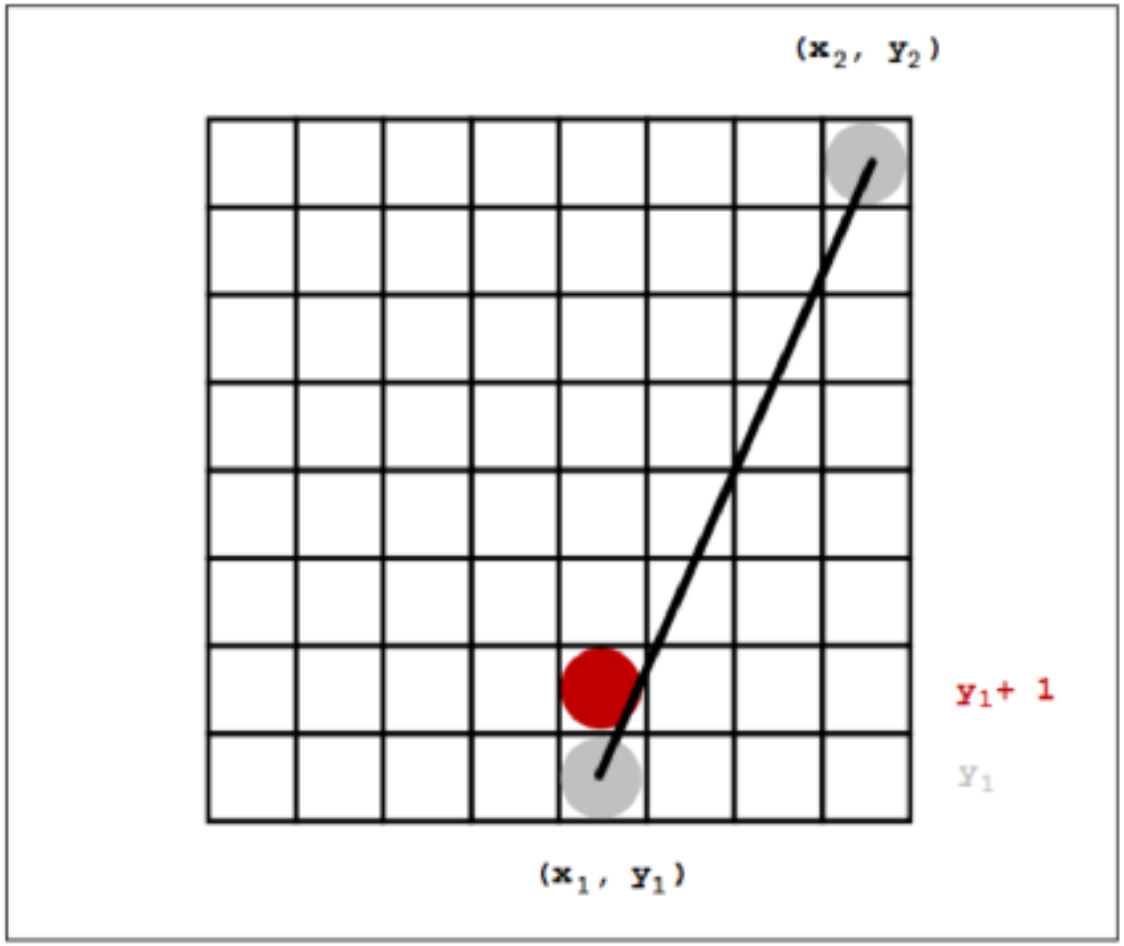


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

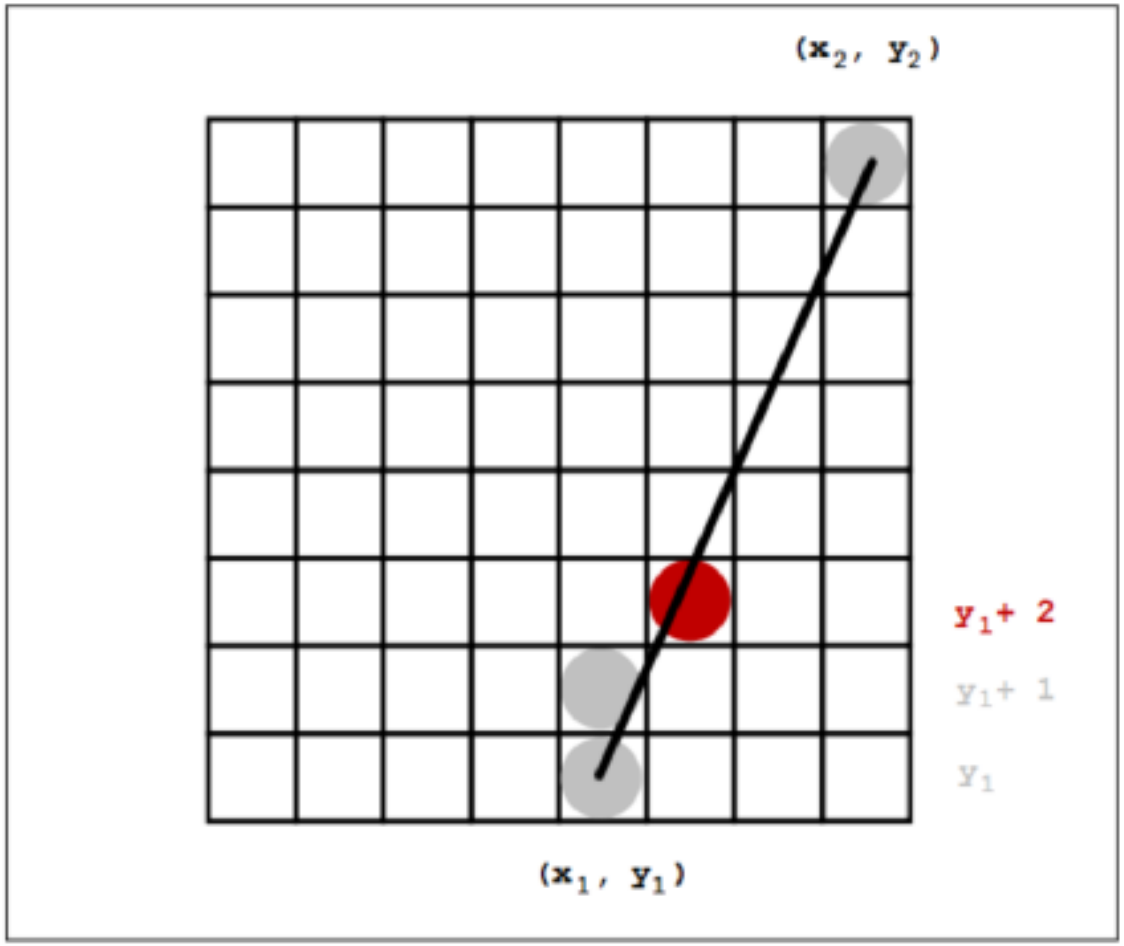


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

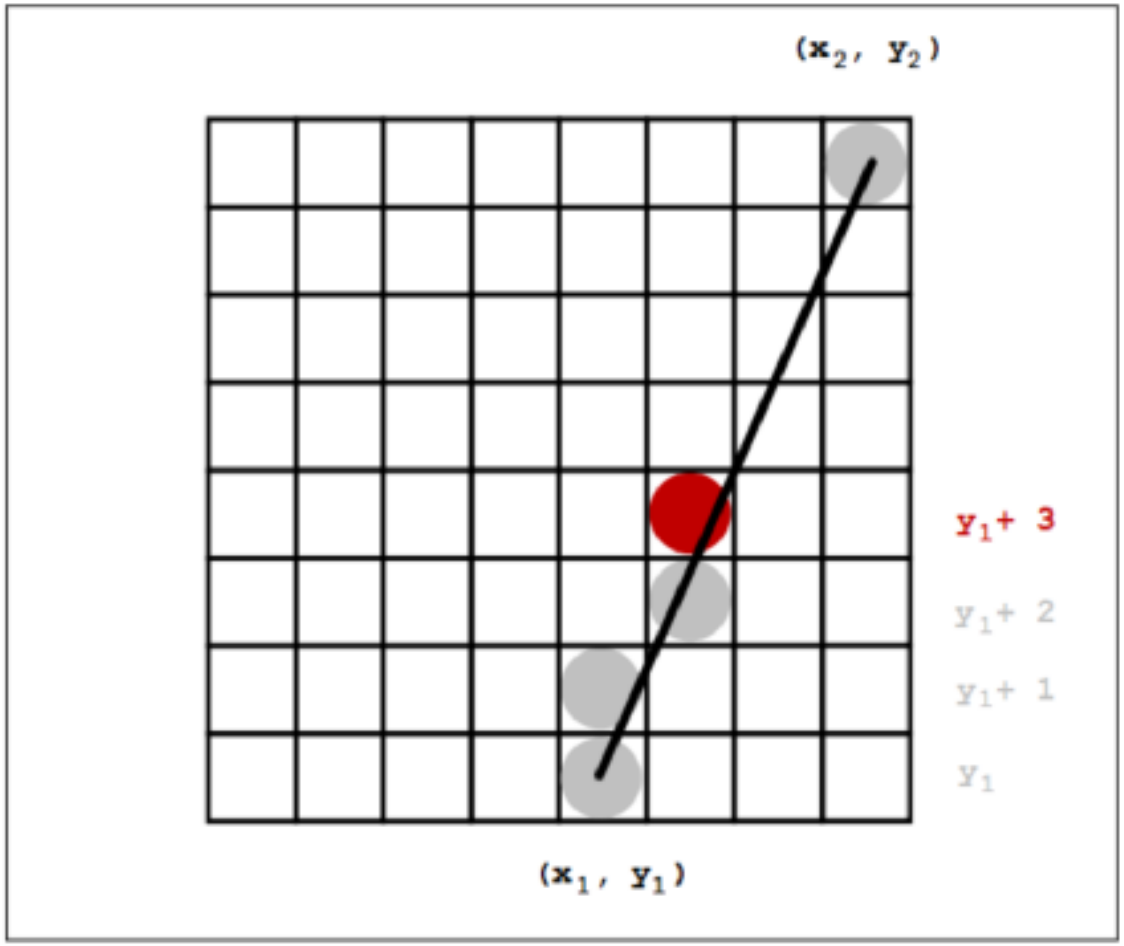


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

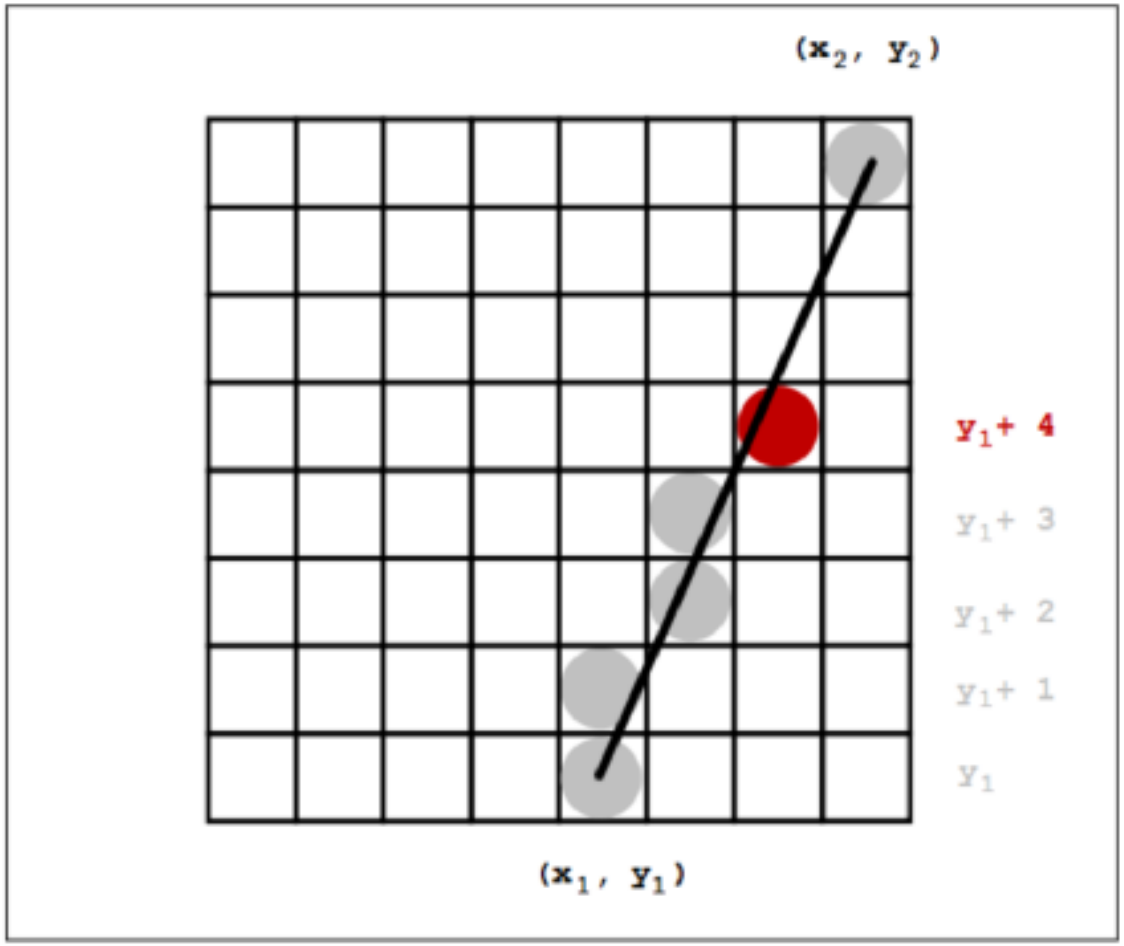


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

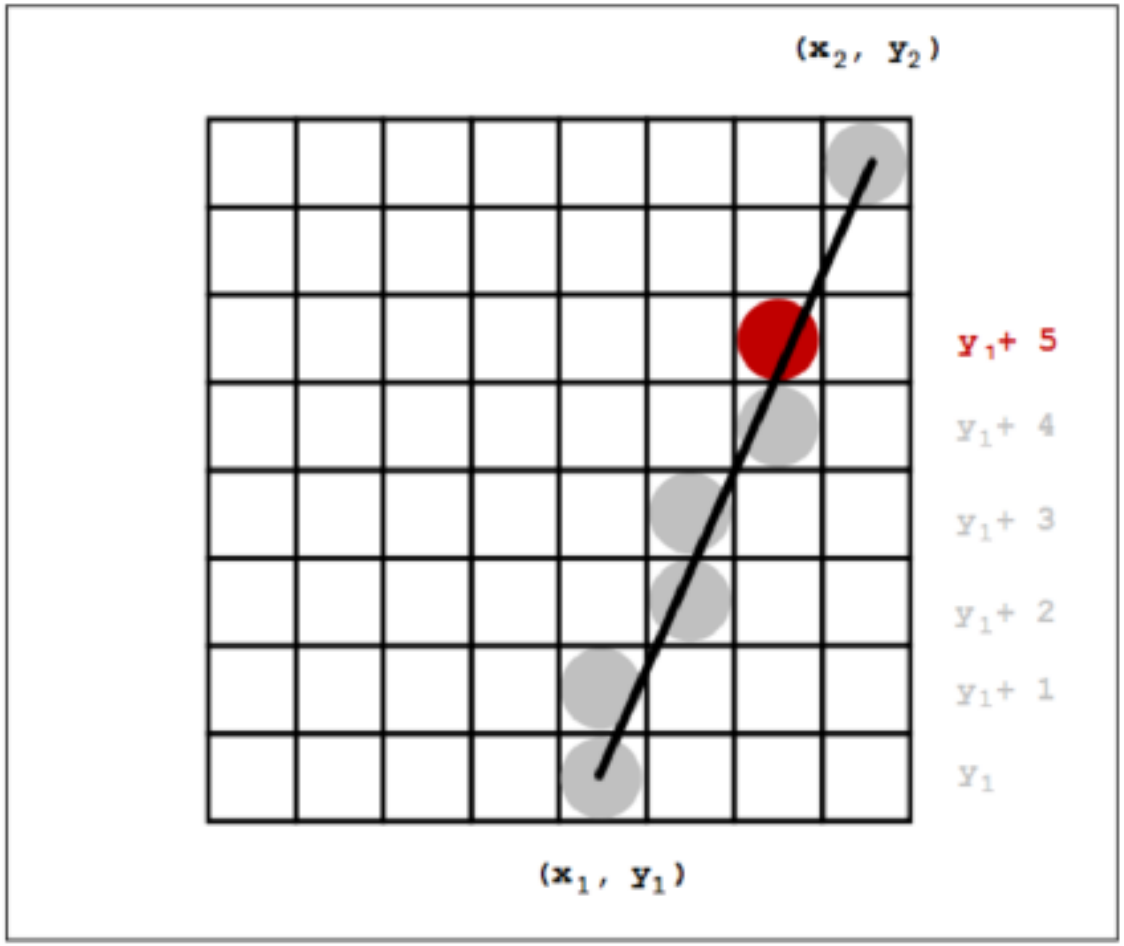


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

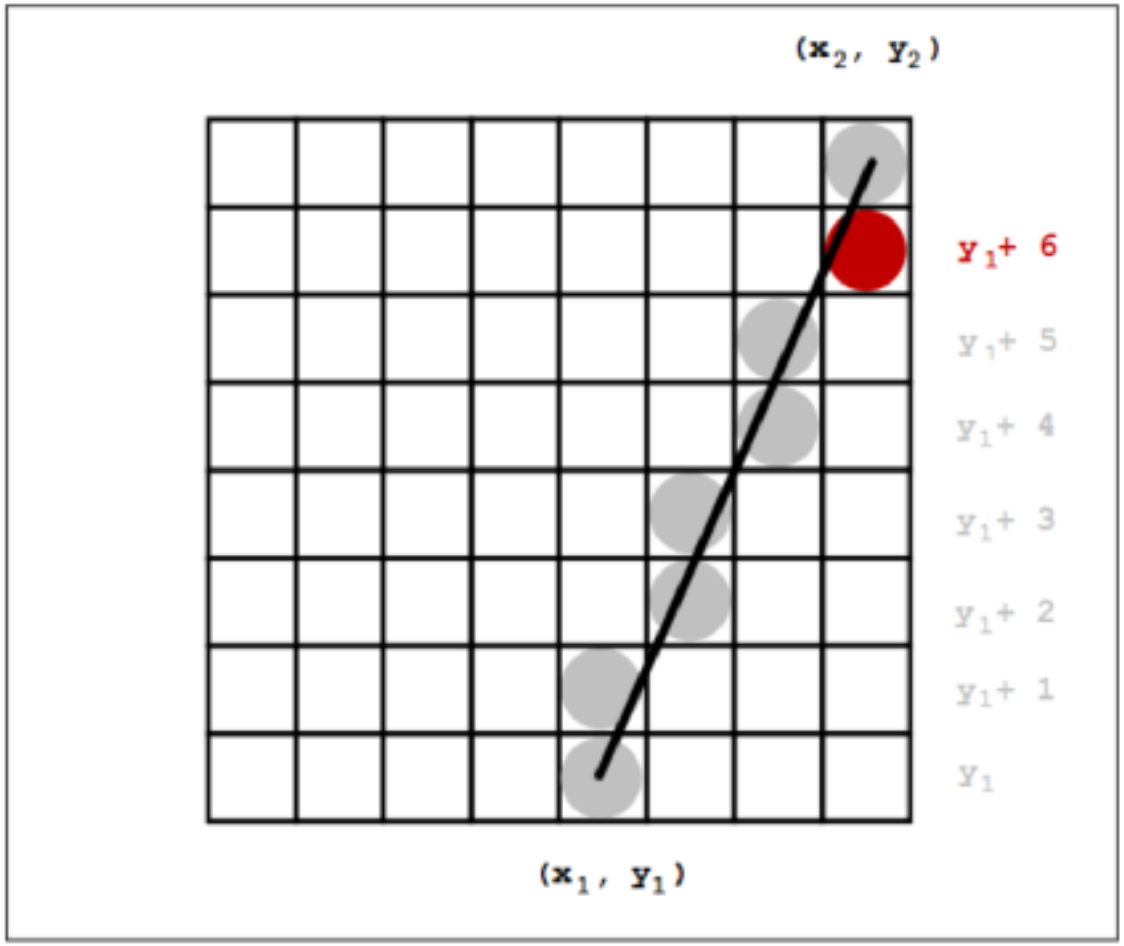


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

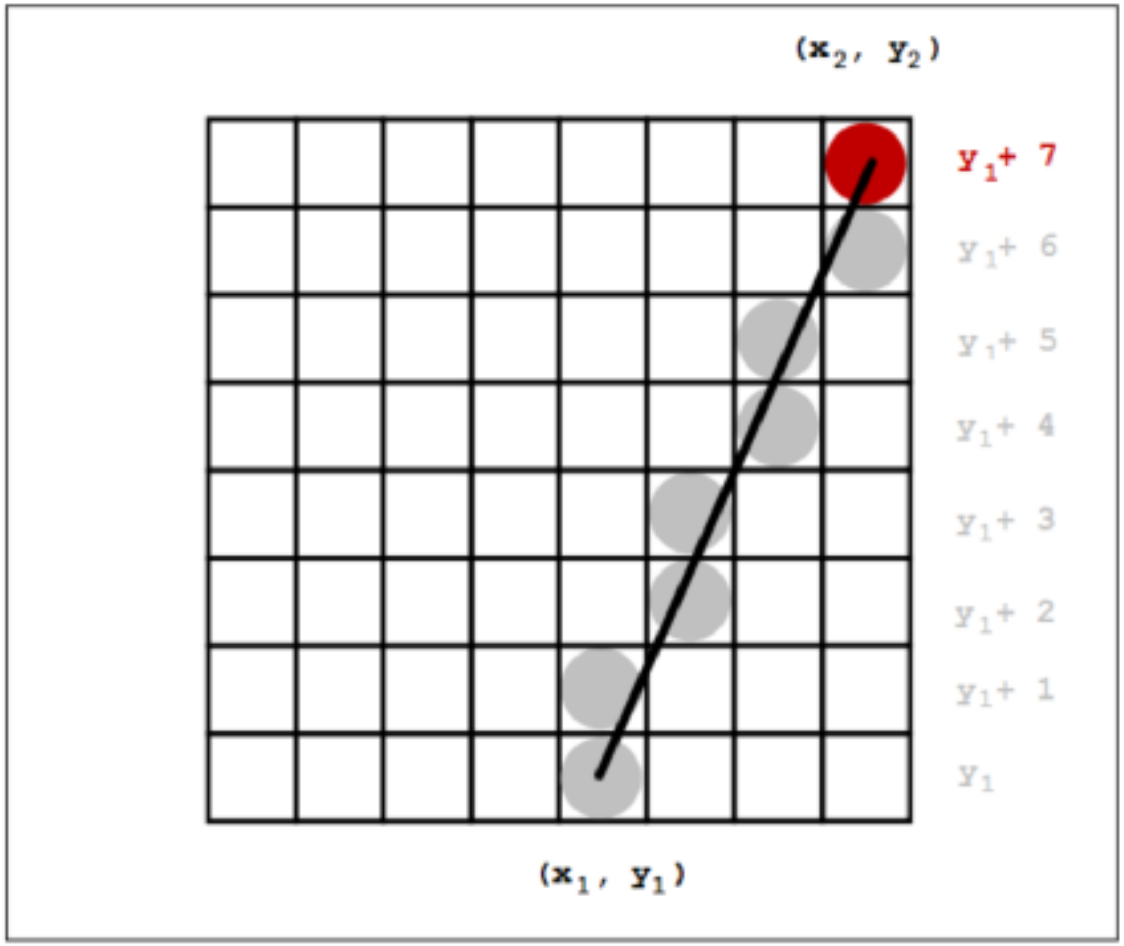


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

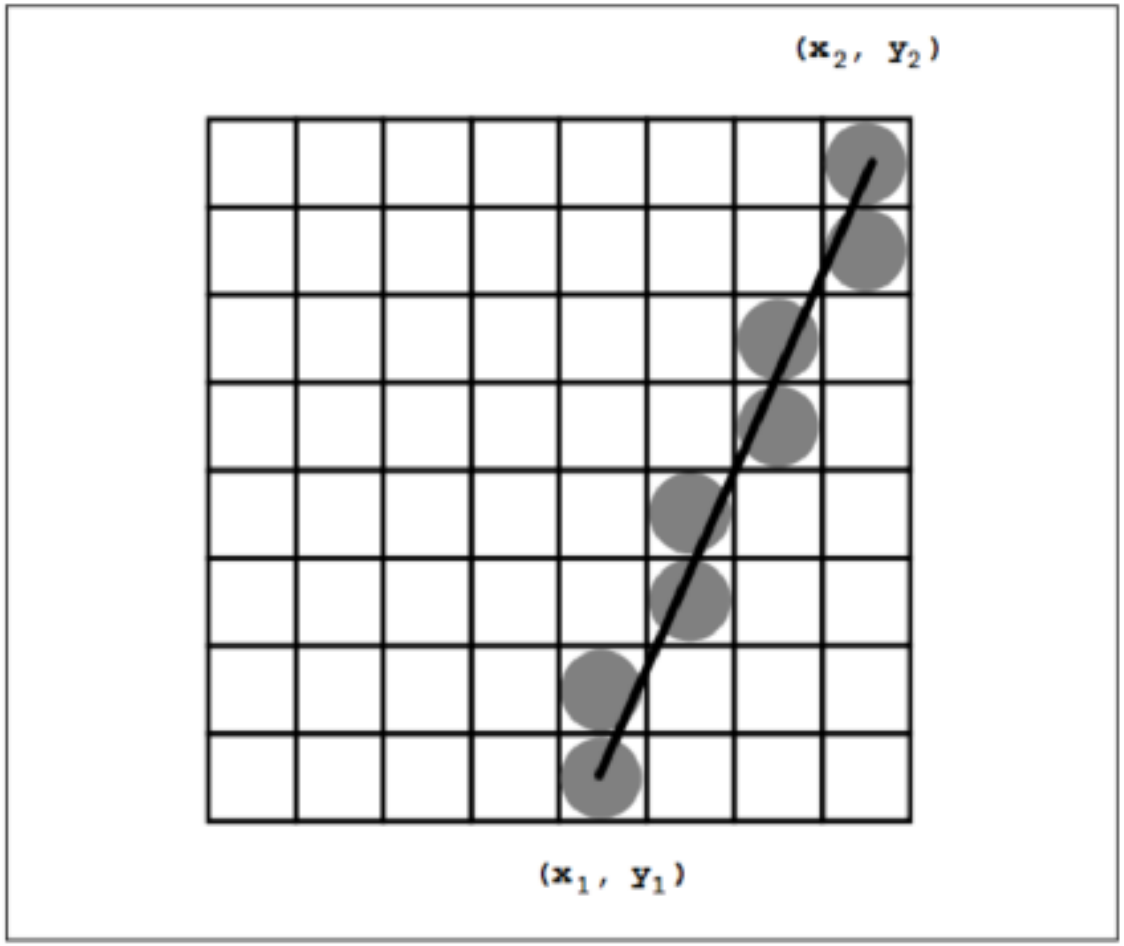


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

Case - II: the algorithm starts from the point with more ordinate value

Example of DDA algorithm when $|m| > 1$

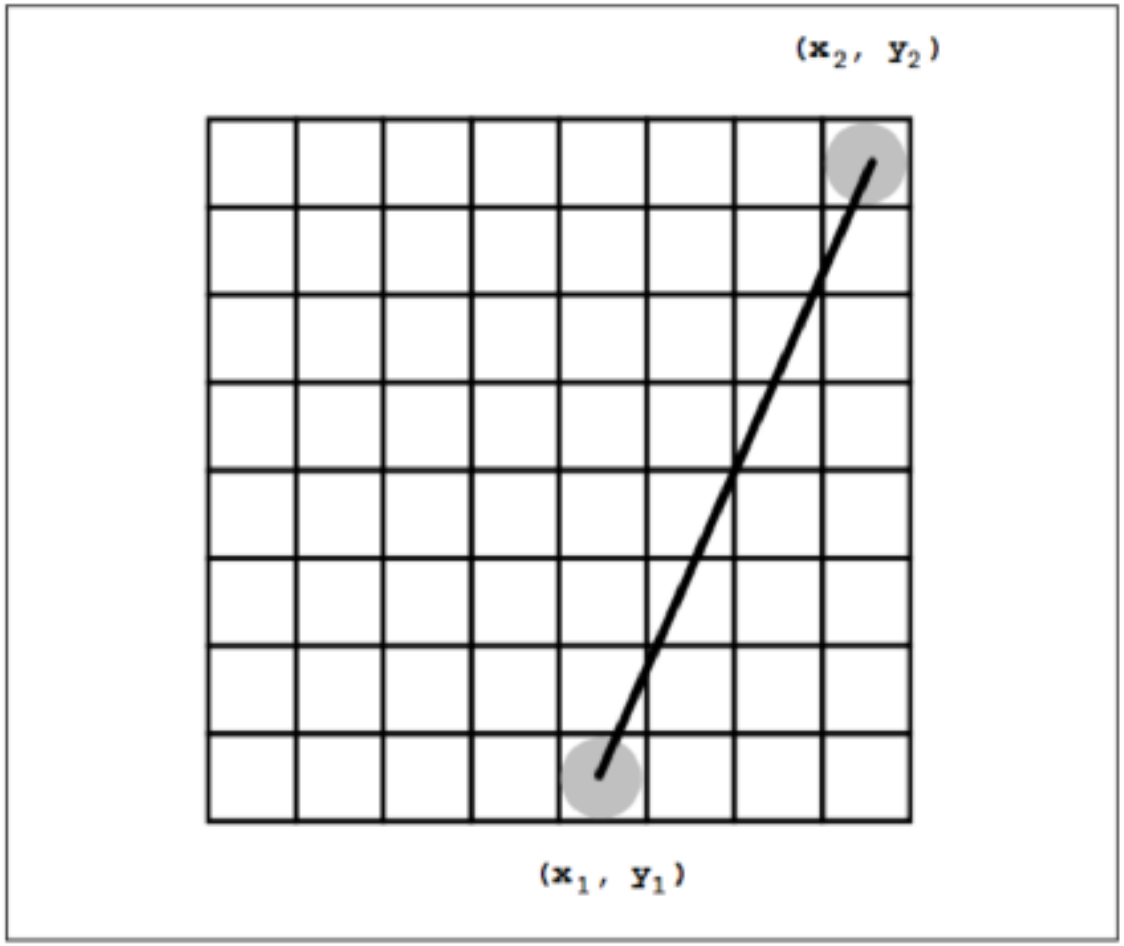


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

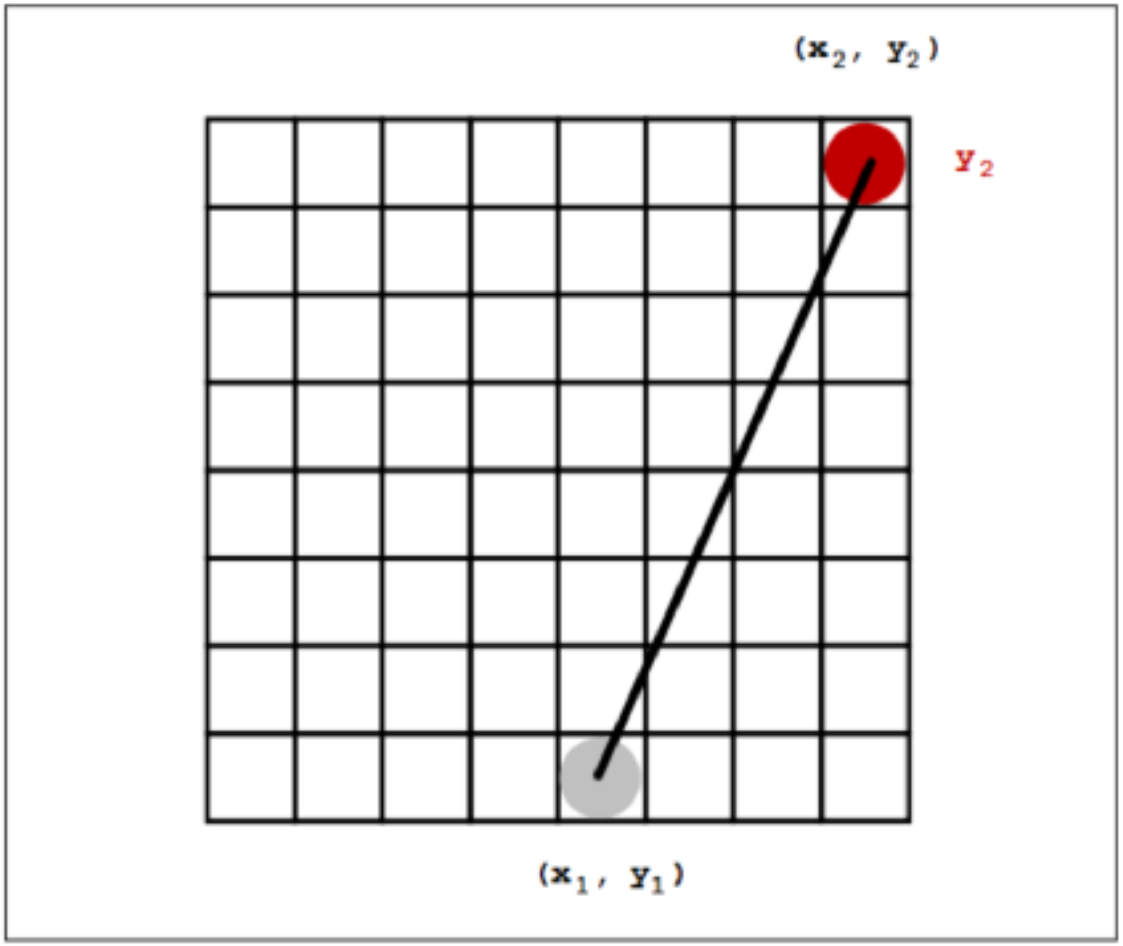


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

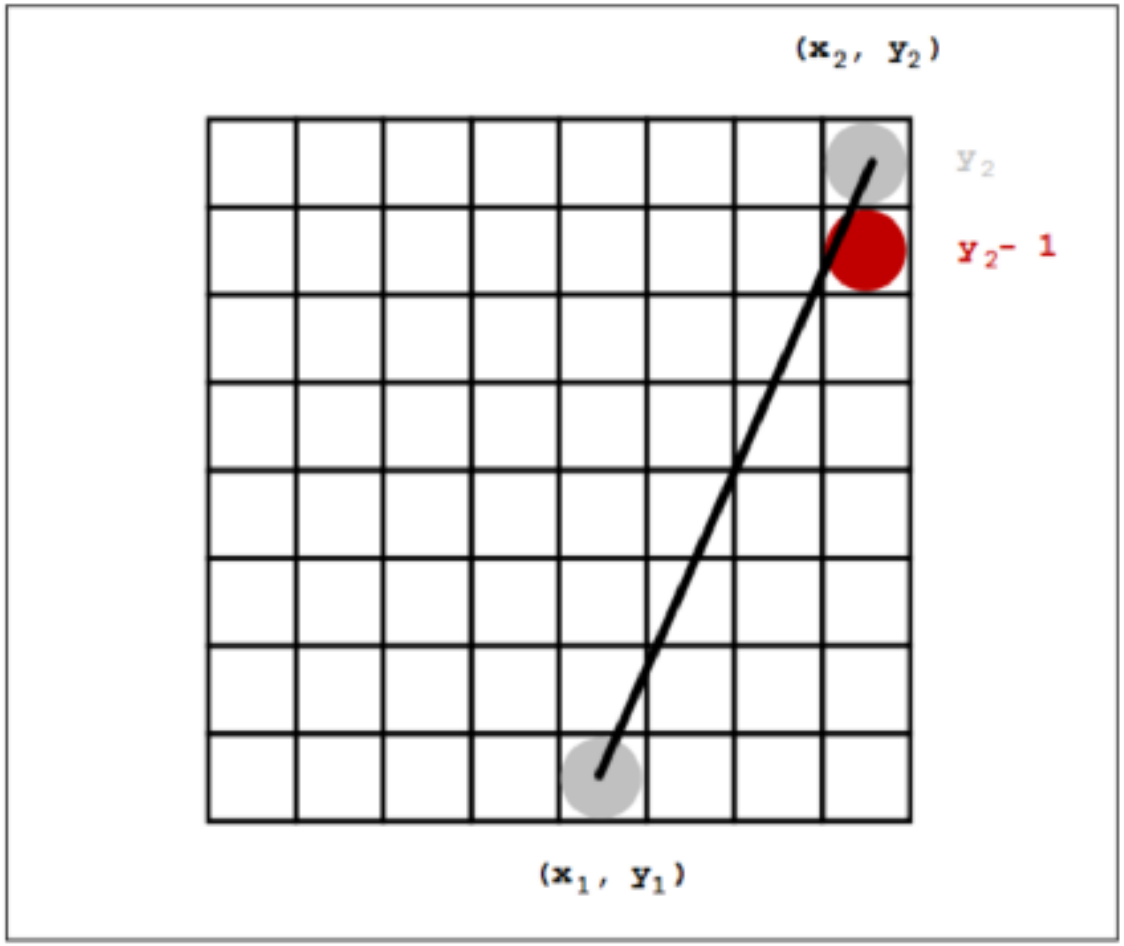


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

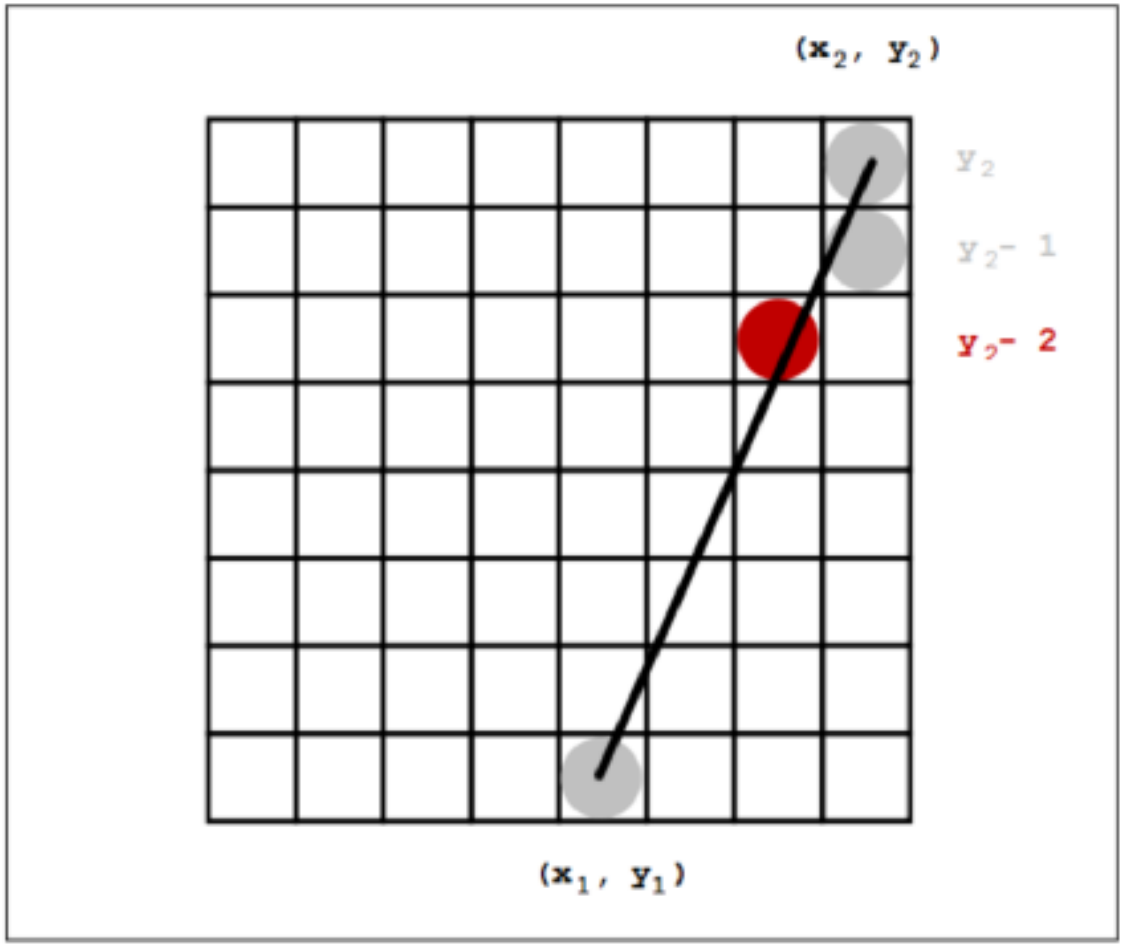


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

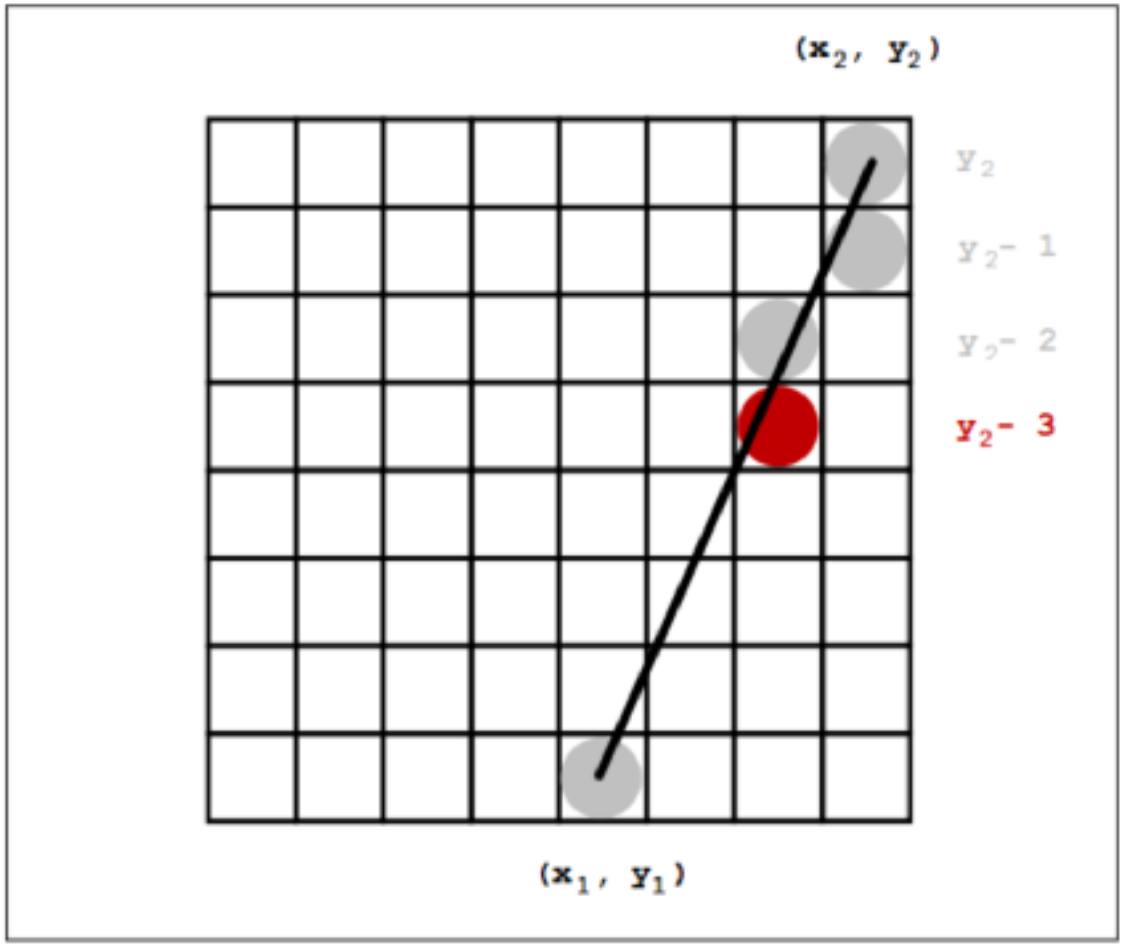


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

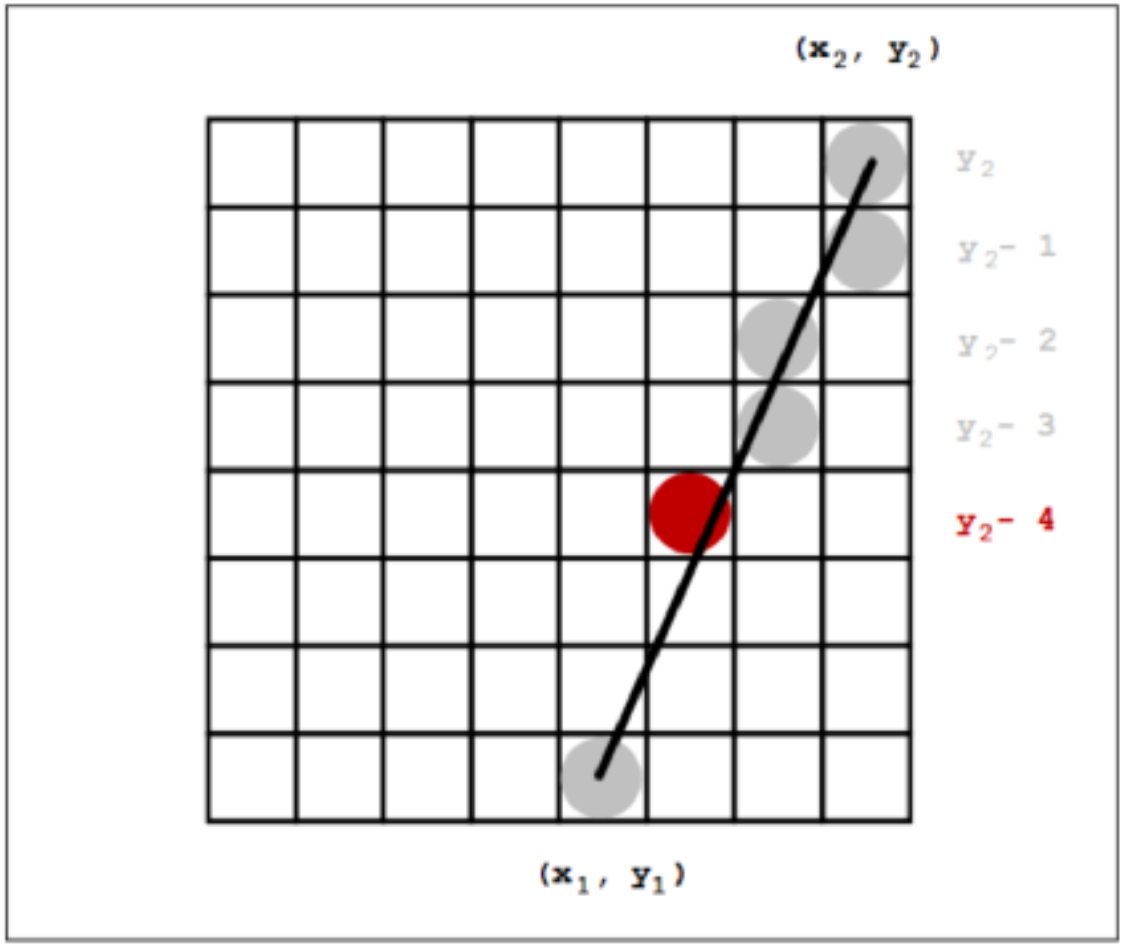


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

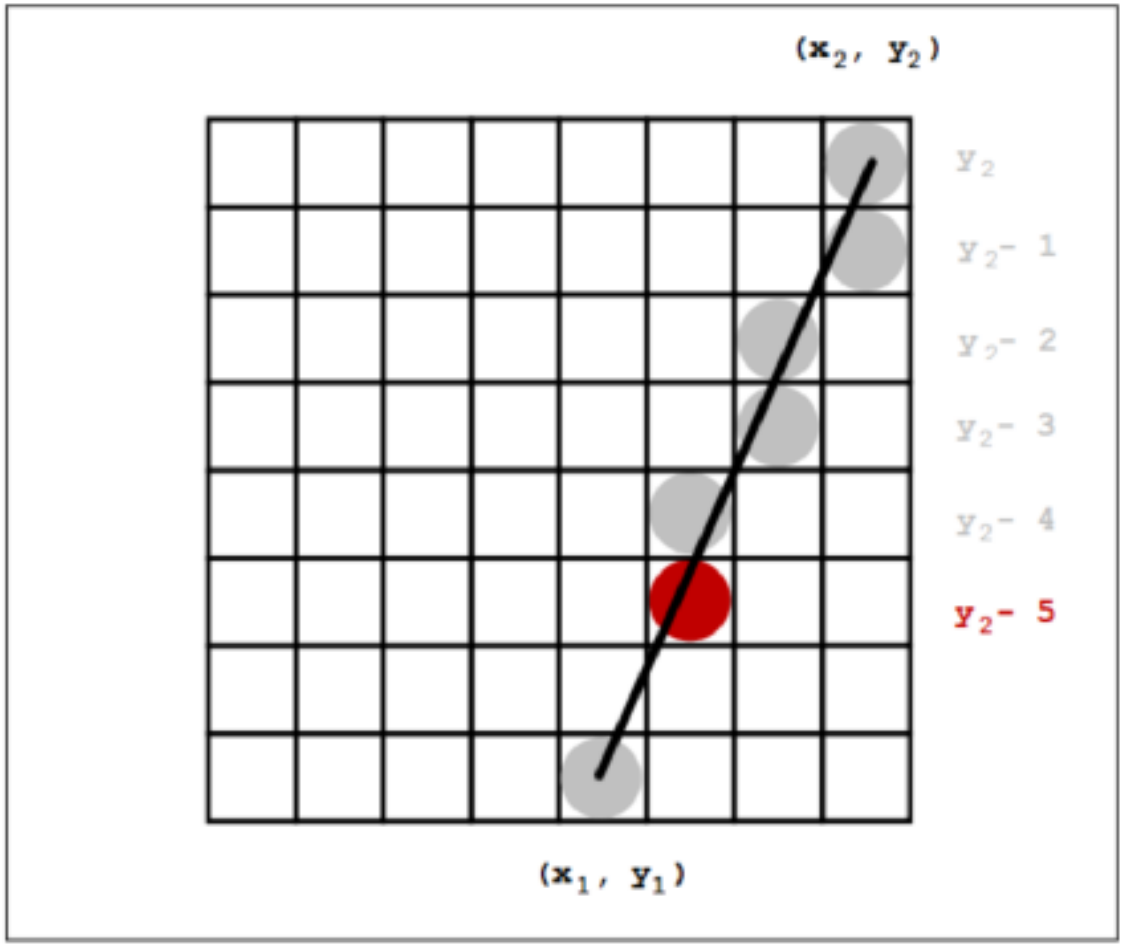


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

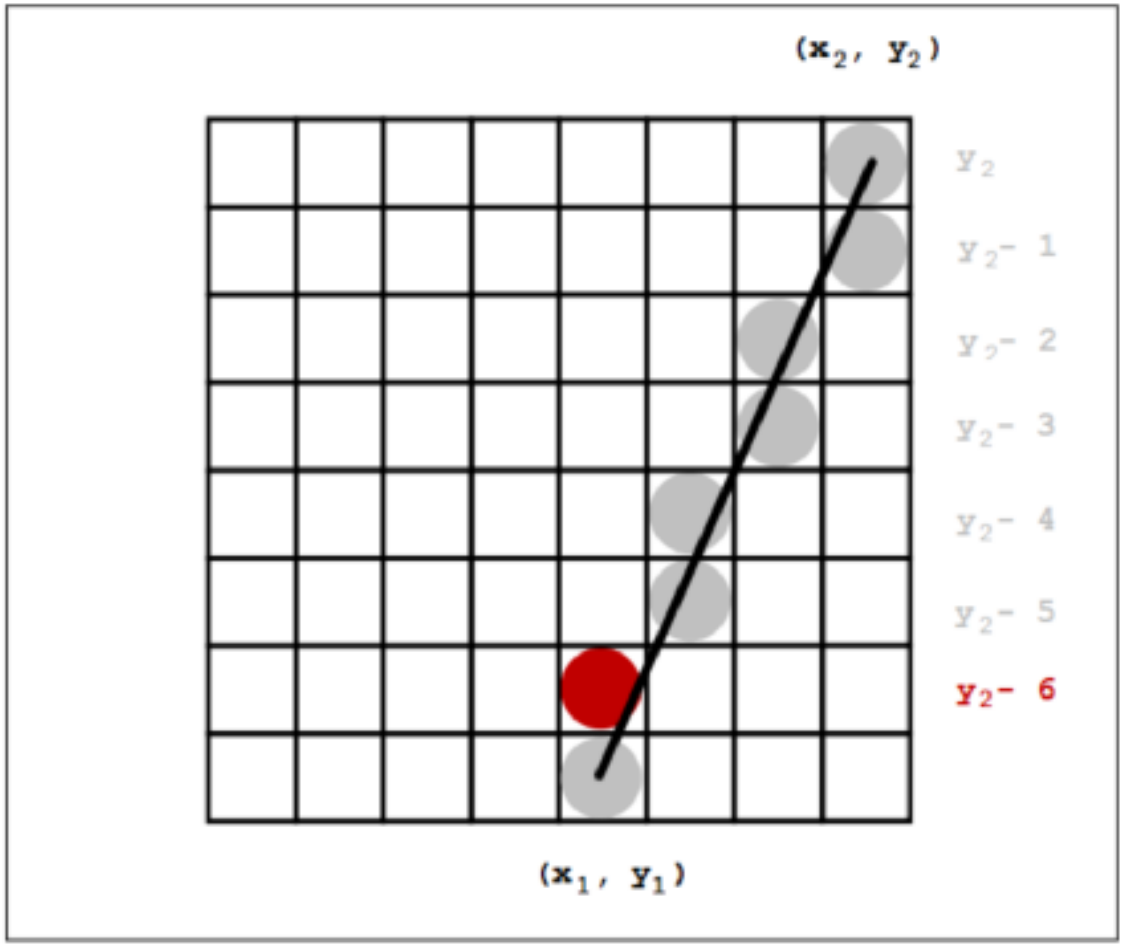


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

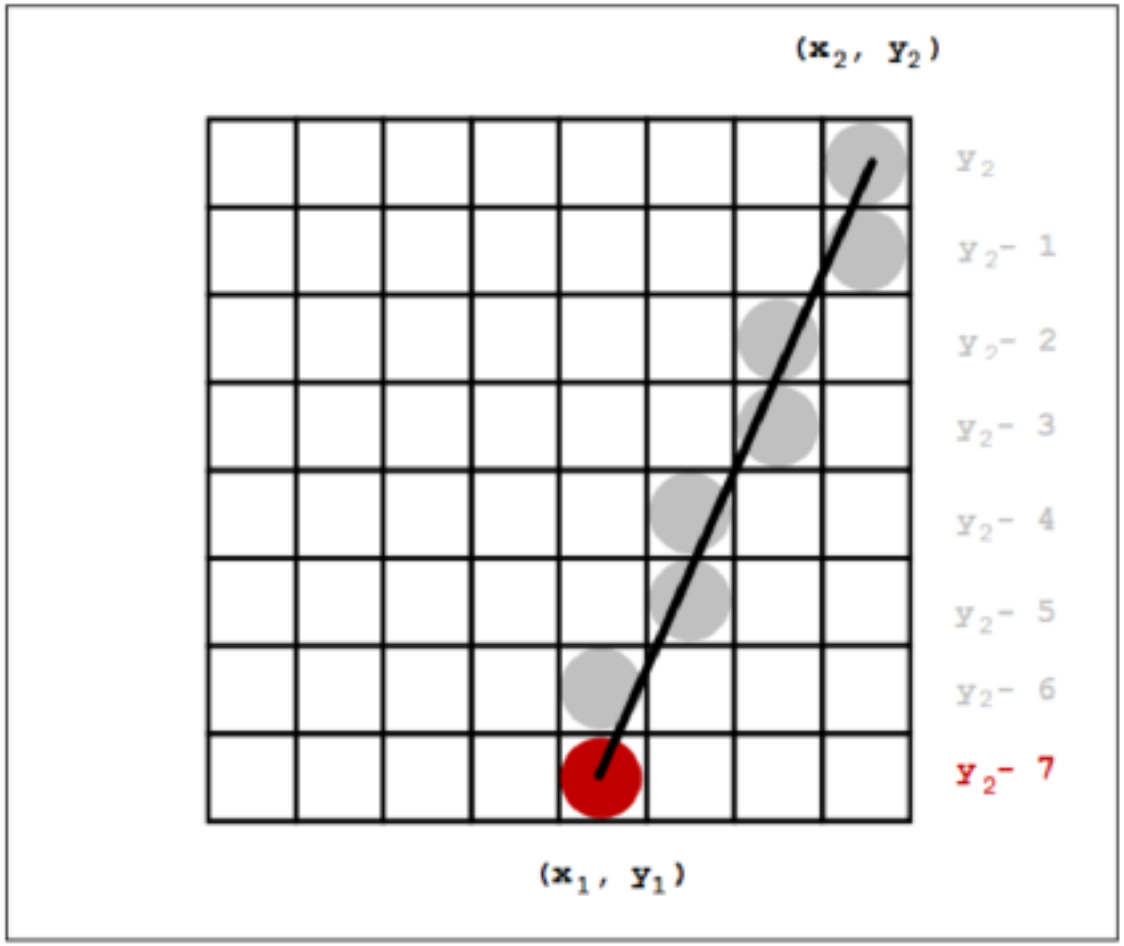


Figure : An example of DDA algorithm when $|m| > 1$

Example of DDA algorithm when $|m| > 1$

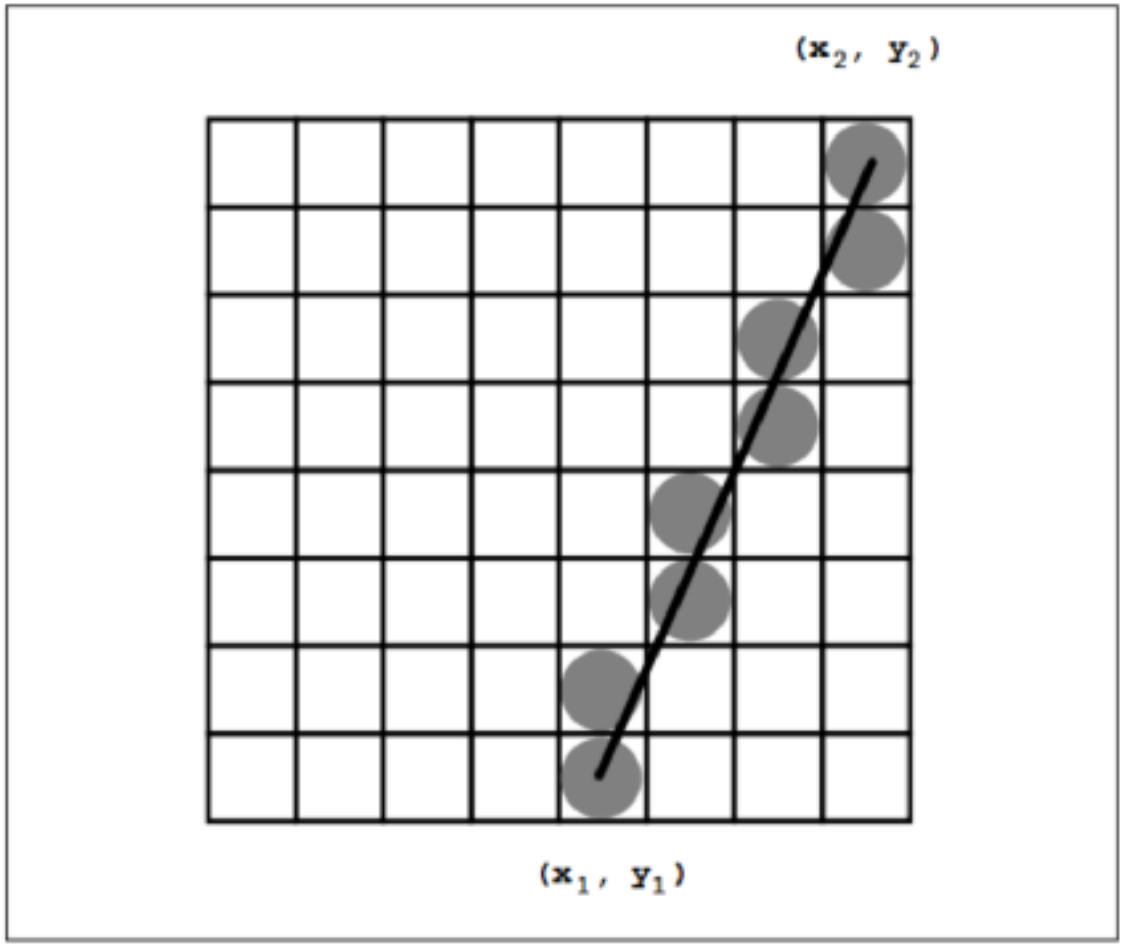


Figure : An example of DDA algorithm when $|m| > 1$

Analysis of DDA algorithm

General line drawing

$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$

Operations needed:
Multiplication: 02
Addition: 04

$m = \frac{y_2 - y_1}{x_2 - x_1}$ can be calculated once and can be stored.

Reduced operation:
Multiplication: 01
Addition: 02

DDA line drawing

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

Operations needed:
Multiplication: 00
Addition: 01

In every step, previous location needs to be stored.

A simple example by DDA

Find the coordinate of raster points on the straight line joining $(20, 10)$ and $(30, 18)$

A simple example by DDA

$m = \frac{18-10}{30-20} = 0.8(< 1) \Rightarrow x$ will be varied incrementally.

x	Actual y ($= y_1$)	y by DDA algorithm (wrong)	y by DDA algorithm ($= y_2$) (wrong?)	y by DDA algorithm ($= y_3$) (correct)
20	10.0	10.0	10.0	10.0
21	10.8	$10 + 0.8 = 10$	$10.0 + 0.8 = 10.8 \approxeq 10$	$10.0 + 0.8 = 10.8 \approxeq 11$
22	11.6	$10 + 0.8 = 10$	$10.8 + 0.8 = 11.6 \approxeq 11$	$10.8 + 0.8 = 11.6 \approxeq 12$
23	12.4	$10 + 0.8 = 10$	$11.6 + 0.8 = 12.4 \approxeq 12$	$11.6 + 0.8 = 12.4 \approxeq 12$
24	13.2	$10 + 0.8 = 10$	$12.4 + 0.8 = 13.2 \approxeq 13$	$12.4 + 0.8 = 13.2 \approxeq 13$
25	14.0	$10 + 0.8 = 10$	$13.2 + 0.8 = 14.0 \approxeq 14$	$13.2 + 0.8 = 14.0 \approxeq 14$
26	14.8	$10 + 0.8 = 10$	$14.0 + 0.8 = 14.8 \approxeq 14$	$14.0 + 0.8 = 14.8 \approxeq 15$
27	15.6	$10 + 0.8 = 10$	$14.8 + 0.8 = 15.6 \approxeq 15$	$14.8 + 0.8 = 15.6 \approxeq 16$
28	16.4	$10 + 0.8 = 10$	$15.6 + 0.8 = 16.4 \approxeq 16$	$15.6 + 0.8 = 16.4 \approxeq 16$
29	17.2	$10 + 0.8 = 10$	$16.4 + 0.8 = 17.2 \approxeq 17$	$16.4 + 0.8 = 17.2 \approxeq 17$
30	18.0	$10 + 0.8 = 10$	$17.2 + 0.8 = 18.0 \approxeq 18$	$17.2 + 0.8 = 18.0 \approxeq 18$

Analyzing the level of error due to rasterization

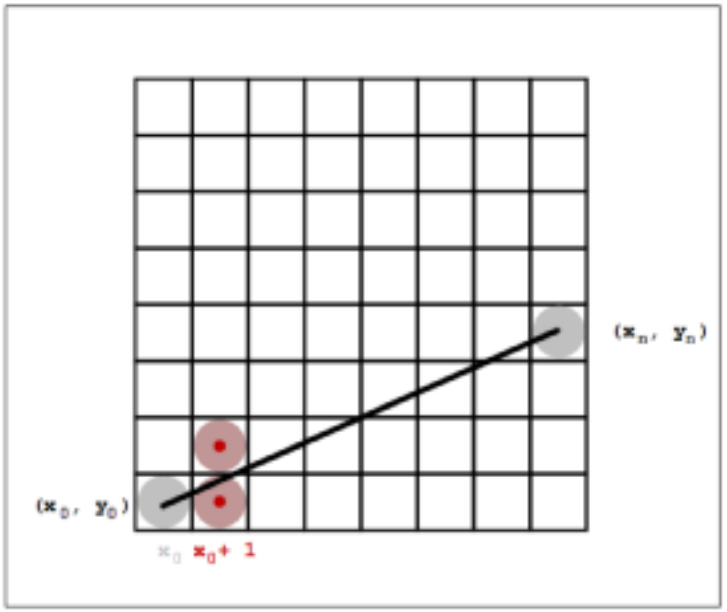
x	20	21	22	23	24	25	26	27	28	29	30
y_1	10	10.8	11.6	12.4	13.2	14.0	14.8	15.6	16.4	17.2	18
y_1'	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	-
y_1''	0	0	0	0	0	0	0	0	0	-	-
y_2	10	10	11	12	13	14	14	15	16	17	18
y_2'	0	1	1	1	1	0	1	1	1	1	-
y_2''	1	0	0	0	-1	1	0	0	0	-	-
y_3	10	11	12	12	13	14	15	16	16	17	18
y_3'	1	1	0	1	1	1	1	0	1	1	-
y_3''	0	-1	1	0	0	0	-1	1	0	-	-

- Number of non-zero element in y_2'' : 3
- Number of non-zero element in y_3'' : 4
- Can this be used for comparing goodness of two outputs? NO
- Can this be used for comparing goodness of plot with own vector version? YES

Bresenham's Line Drawing Algorithm

Introduction from equation of a line

- Cartesian *slope-intercept equation* for a straight line:
  $y = m.x + b$
- Given a point (x_k, y_k), we want to decide what will be the value of (x_{k+1}, y_{k+1})?



Why a decision parameter?

- When $|m| \leq 1$:
  $x_{k+1} = x_k + 1$
  $y_{k+1} = y_k$ OR $y_{k+1} = y_k + 1$ (based on decision parameter)
- When $|m| > 1$:
  $y_{k+1} = y_k + 1$
  $x_{k+1} = x_k$ OR $x_{k+1} = x_k + 1$ (based on decision parameter)

Significance of decision parameter

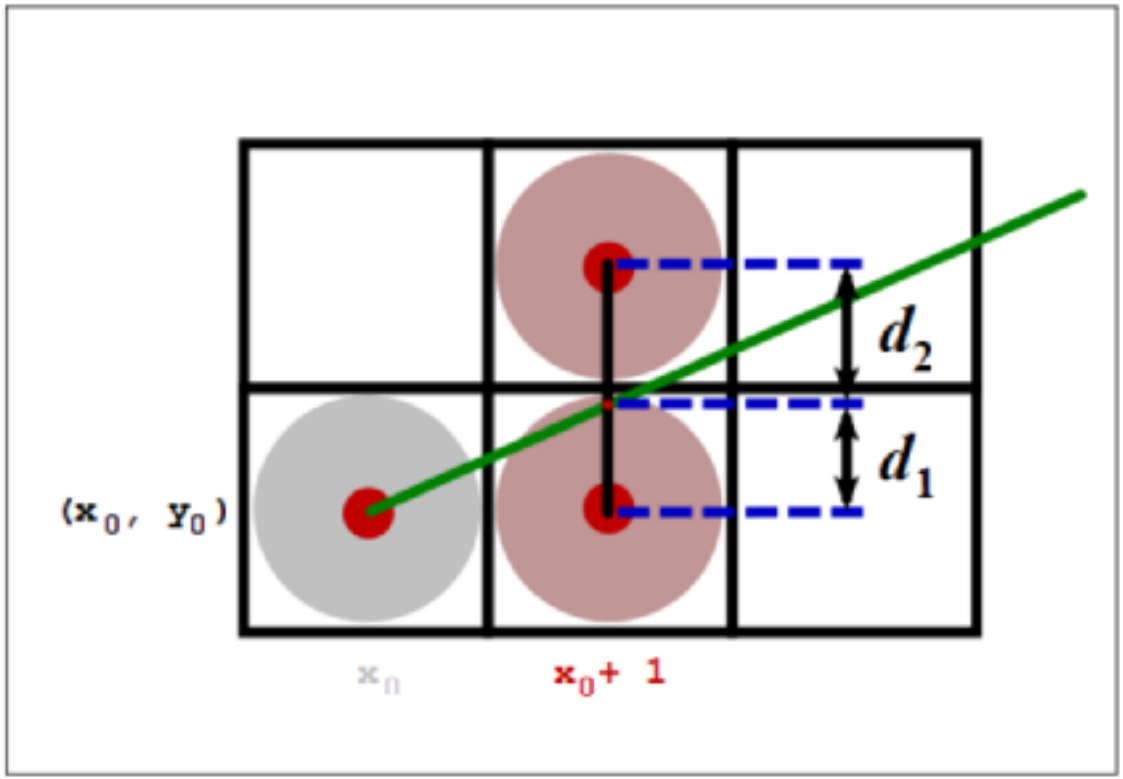


Figure : Significance of decision parameter

Formation of the decision parameter

- $y_k = mx_k + b \Rightarrow y_{k+1} = m(x_{k+1}) + b$
- For $|m| \leq 1$, $x_{k+1} = x_k + 1$
- We define d_1 as:
  $d_1 = y_{k+1} - y_k = m(x_k + 1) + b - y_k$
- We define d_2 as:
  $d_2 = (y_k + 1) - y_{k+1} = y_k + 1 - m(x_k + 1) - b$
- $d_1 - d_2 = 2m(x_k + 1) - 2y_k + 2b - 1$

Formation of the decision parameter

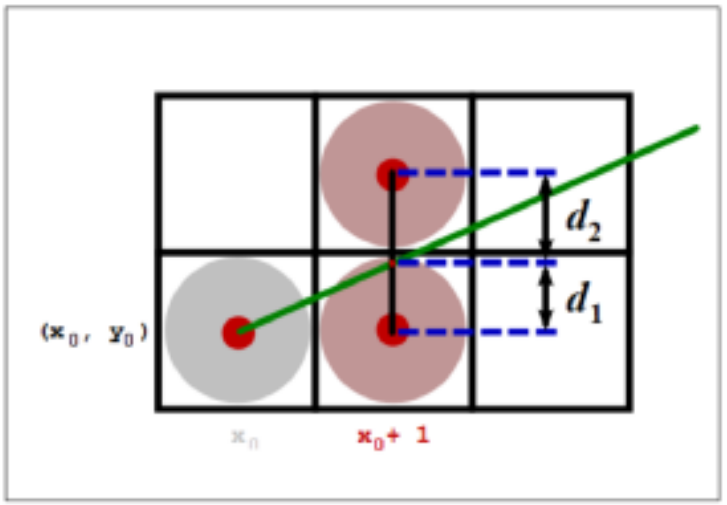
- Decision Parameter p_k
  $= \Delta x(d_1 - d_2)$
  $= \Delta x[2m(x_k + 1) - 2y_k + 2b - 1]$
  $= 2\Delta y.x_k - 2\Delta x.y_k + c$
  where, $m = \frac{\Delta y}{\Delta x}$, and $c = 2\Delta y + \Delta x(2b - 1)$
- c is a constant

Formation of the decision parameter

- $p_k = function(x_k, y_k) = 2\Delta y.x_k - 2\Delta x.y_k + c$
- $p_{k+1} = function(x_{k+1}, y_{k+1}) = 2\Delta y.x_{k+1} - 2\Delta x.y_{k+1} + c$
- $p_{k+1} - p_k = 2\Delta y(x_{k+1} - x_k) - 2\Delta x(y_{k+1} - y_k)$
- We already know: for $|m| \leq 1$, $x_{k+1} = x_k + 1$
- $p_{k+1} = p_k + 2\Delta y - 2\Delta x(y_{k+1} - y_k)$ $(\forall k \geq 0)$
- $p_0 =???$

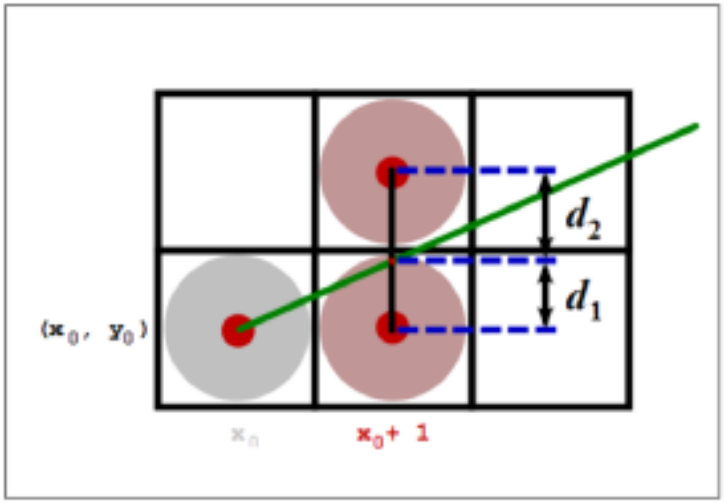
Finding p_0

- p_0
  $= \Delta x(d_1 - d_2)$
  $= \Delta x(2d_1 - 1)$
  (as $d_1 + d_2 = 1$)
- Slope (m)
  $= \frac{\Delta y}{\Delta x}$ (in general)
  $= \frac{d_1}{1}$
- p_0
  $= \Delta x(2m - 1)$
  $= \Delta x(2\frac{\Delta y}{\Delta x} - 1)$
  $= 2\Delta y - \Delta x$



Formation of the decision parameter

- Decision Parameter
  $p_k = \Delta x(d_1 - d_2)$ where $\Delta x > 0$
- Sign of p_k depends on sign of $(d_1 - d_2)$
- $p_{k+1} =$
  $p_k + 2\Delta y - 2\Delta x(y_{k+1} - y_k)$
  $(\forall k \geq 0)$
- IF $p_k < 0$,
  $d_2 > d_1 \Rightarrow y_{k+1} = y_k$
  $p_{k+1} = p_k + 2\Delta y$
- OTHERWISE,
  $d_1 > d_2 \Rightarrow y_{k+1} = y_k + 1$
  $p_{k+1} = p_k + 2\Delta y - 2\Delta x$



Bresenham's line drawing algorithm with $|m| \leq 1$

1. Input: (x_0, y_0) and (x_n, y_n) such that $|m| \leq 1$
   (assume $x_0 < x_n$ without loss of generality)
2. Calculate $\Delta x, \Delta y, 2\Delta y$, and $(2\Delta y - 2\Delta x)$
3. Load (x_0, y_0)
4. $p_0 \leftarrow (2\Delta y - \Delta x)$
5. $k \leftarrow 0$
6. if $(p_k < 0)$
   Load $(x_k + 1, y_k)$
   $p_{k+1} \leftarrow p_k + 2\Delta y$
7. Otherwise
   Load $(x_k + 1, y_k + 1)$
   $p_{k+1} \leftarrow p_k + (2\Delta y - 2\Delta x)$
8. $k \leftarrow (k + 1)$
9. Repeat steps 6,7,8 for Δx times

A simple example by BLD

Find the coordinate of raster points on the straight line joining $(20, 10)$ and $(30, 18)$

A simple example by BLD

$m = \frac{18-10}{30-20} = 0.8\ (< 1)$
$\Rightarrow x$ will be varied incrementally.
$\Delta x = 30 - 20 = 10,$
$\Delta y = 18 - 10 = 8,$
$2\Delta y = 16,$
and $(2\Delta y - 2\Delta x) = -4$

k	p_k	x_{k+1}	y_{k+1}
0	6	21	11
1	2	22	12
2	-2	23	12
3	14	24	13
4	10	25	14
5	6	26	15
6	2	27	16
7	-2	28	16
8	14	29	17
9	10	30	18

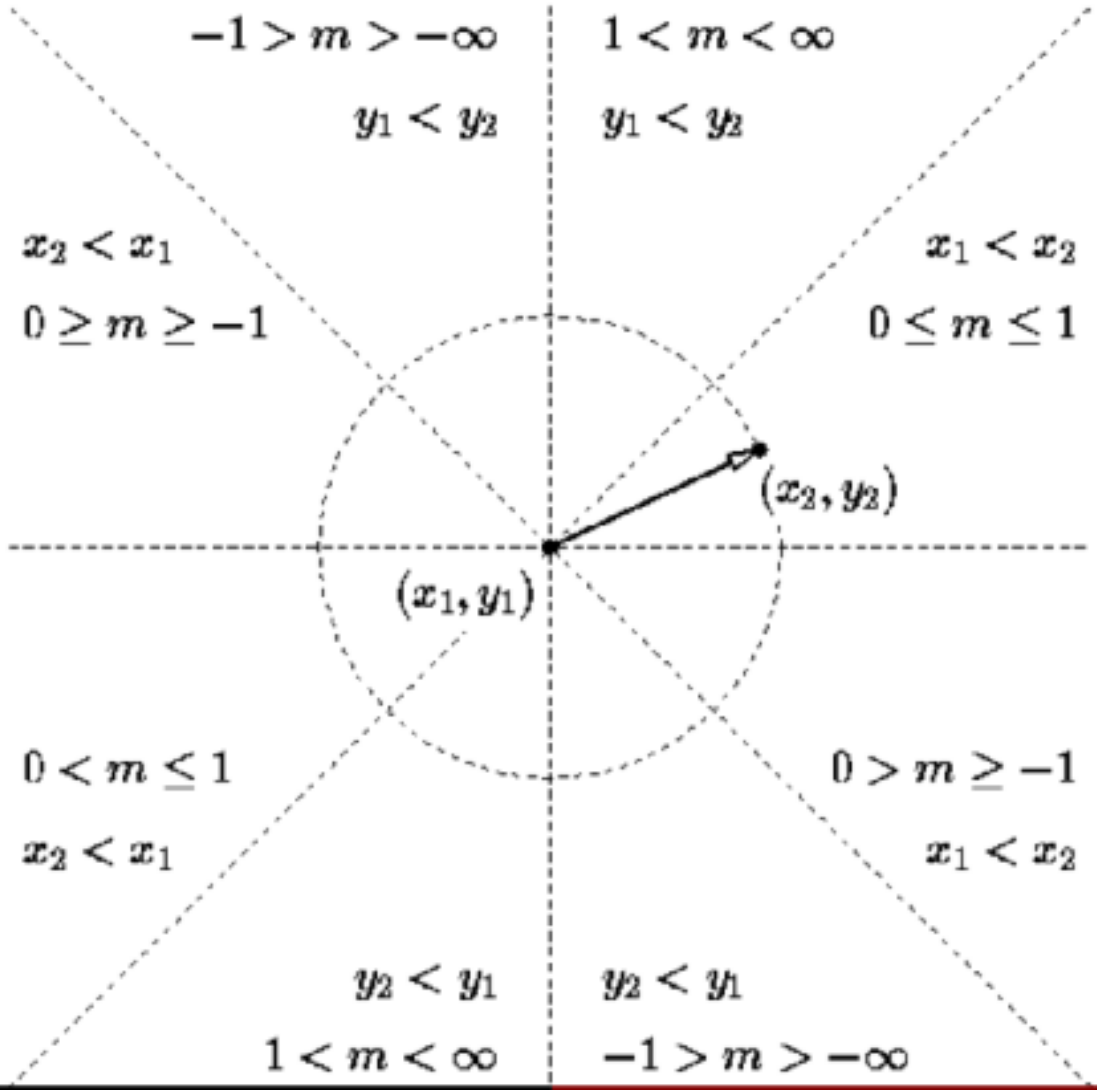
Comparing DDA and BLD on this example

x	y by DDA	y by BDS
20	10	10
21	11	11
22	12	12
23	12	12
24	13	13
25	14	14
26	15	15
27	16	16
28	16	16
29	17	17
30	18	18

Why understanding an algorithm for $|m| < 1$ is sufficient?

- DDA and BLD are sequential line drawing algorithms
- They decide on position of points one by one
- As position of k th point depends on position of $(k-1)$th point, we need to evaluate $(k-1)$ th point before k th.
- This nature restricts the algorithms to exploit parallelism (concurrent plotting of points)

Why understanding an algorithm for $|m| < 1$ is sufficient?



Why understanding an algorithm for $|m| < 1$ is sufficient?

- Reflection along lines $x = 0$, $y = 0$, $x = y$, and $x = -y$ are less computation intensive
- We can find a reflected version of every given line where $|m| < 1$ holds good for reflected version

How to choose the axis for iteration?

- We choose to vary x axis when $|m| \leq 1$
- We choose to vary y axis when $|m| > 1$
- (WHY?)

How to choose the axis for iteration?

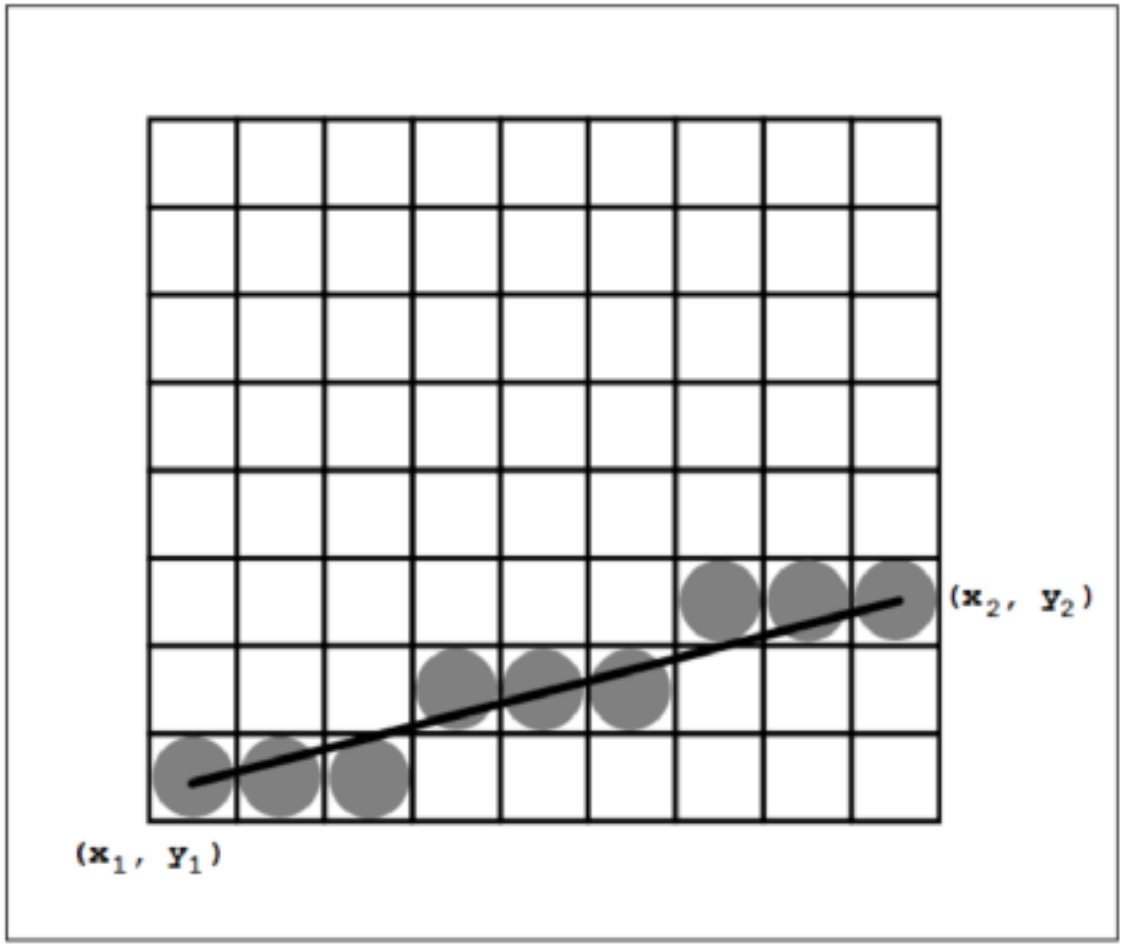


Figure : Going through x axis when $|m| \leq 1$

How to choose the axis for iteration?

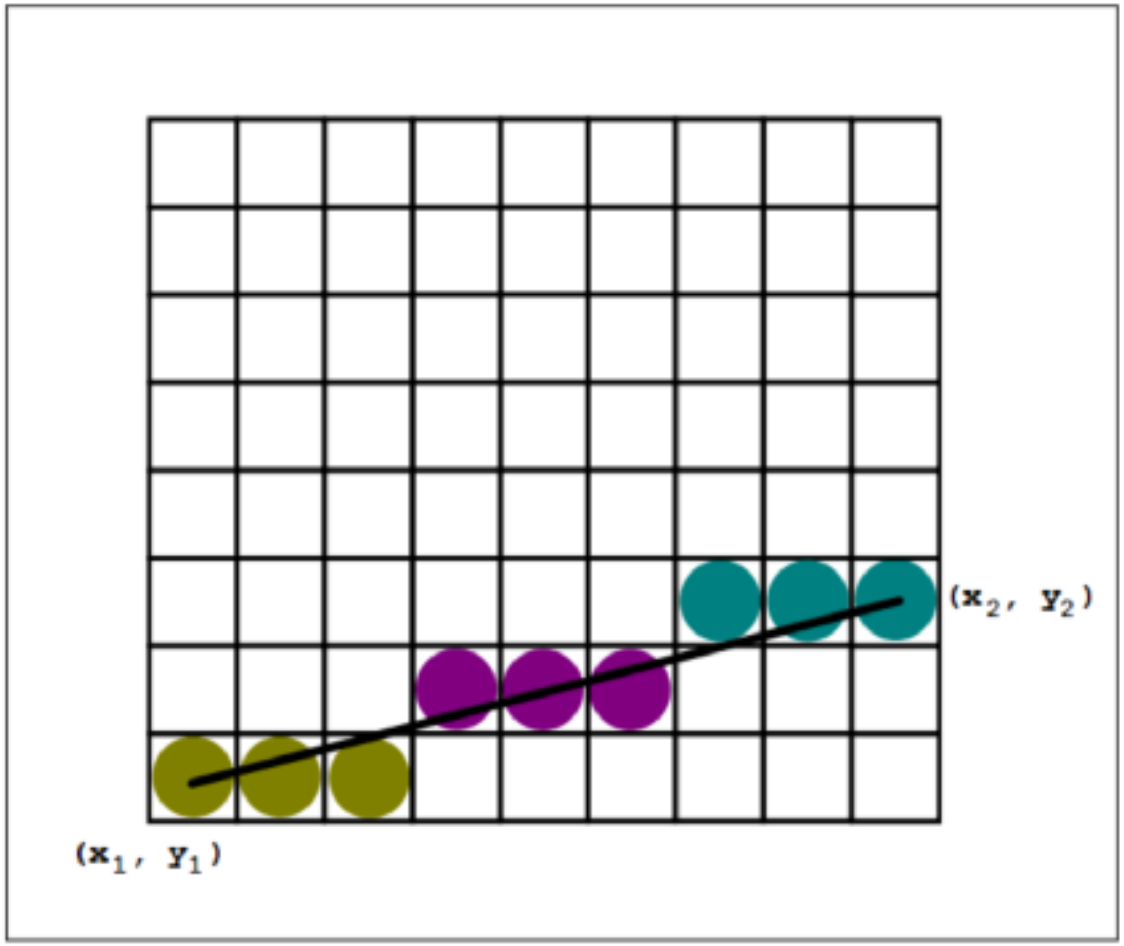


Figure : Going through y axis when when $|m| \leq 1$

How to choose the axis for iteration?

- Constraint 1: To generate maximum number of points with membership to the line
- Constraint 2: Not neglecting any point with membership to the line
- Constraint 3: Maintaining the 8-neighbourhood connectivity

How to choose the axis for iteration?

- Constraint 1: To generate maximum number of points with membership to the line
- Constraint 2: Not neglecting any point with membership to the line
- Constraint 3: Maintaining the 8-neighbourhood connectivity

How to draw two parallel lines?

- We need to draw two lines
- one between (x_1, y_1) and (x_2, y_2), and another between (x_3, y_3) and (x_4, y_4)
- such that $\frac{(y_2-y_1)}{(x_2-x_1)} = \frac{(y_4-y_3)}{(x_4-x_3)}$

How to draw two parallel lines?

- Draw line between (x_1, y_1) and (x_2, y_2) with any algorithms
- Learn the decision pattern
- Apply the decision pattern on line between (x_3, y_3) and (x_4, y_4)
- You need not recompute the decision parameters, and can save time