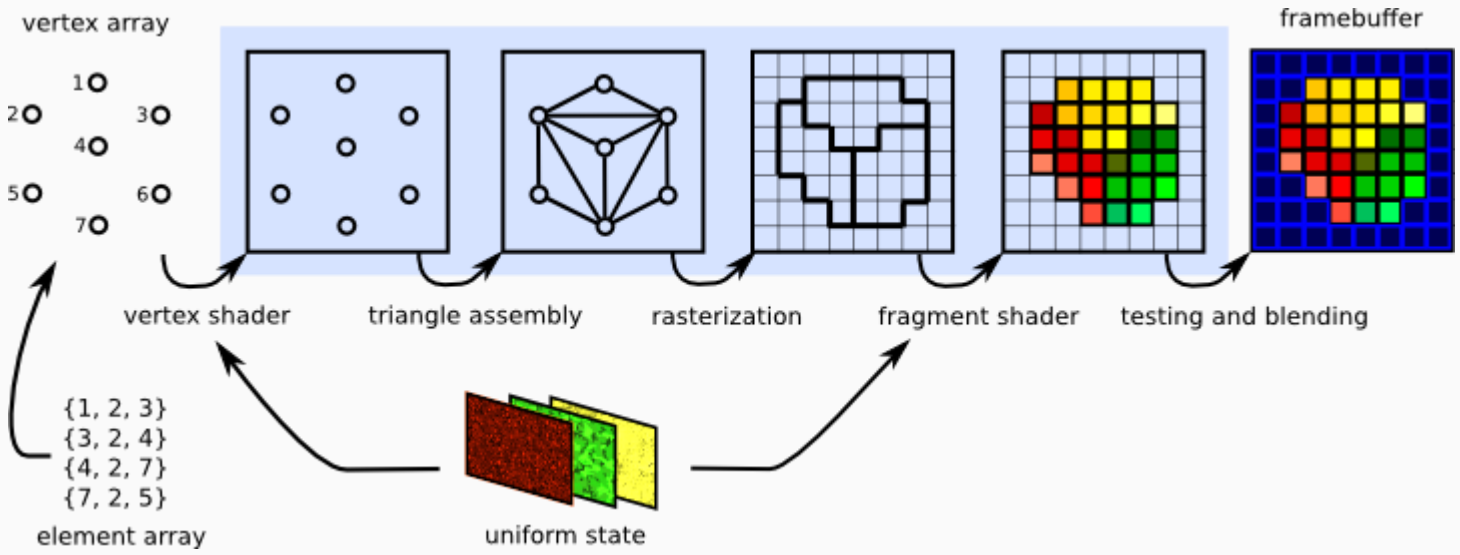
CS113 - ĐỒ HỌA MÁY TÍNH VÀ XỬ LÝ ẢNH

Rasterization – Phần 1 : Đoạn thẳng, đường tròn

Quy trình hiển thị



- Trong bước rời rạc hóa (Rasterization), ta cần xác định các điểm trên lưới pixel (pixel-grid) tương ứng với đối tượng cần được hiển thị.

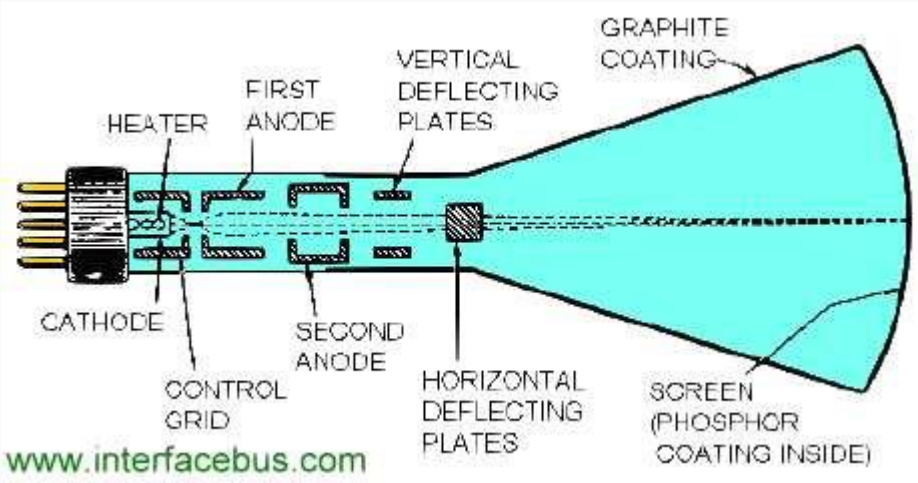
Rasterization

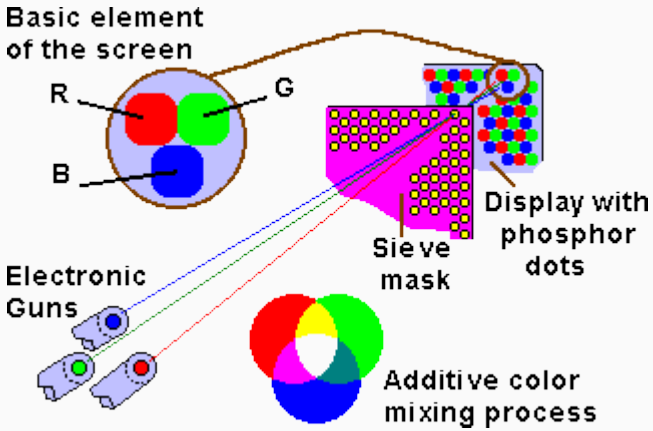
Rasterization (scan conversion)

- Determine which pixels are inside primitive specified by a set of vertices.
- This produces a set of fragments.
- Fragments have a location (pixel location) and other attributes such color and texture coordinates that are determined by interpolating values at vertices.

Pixel colors determined later using color, texture, and other vertex properties.

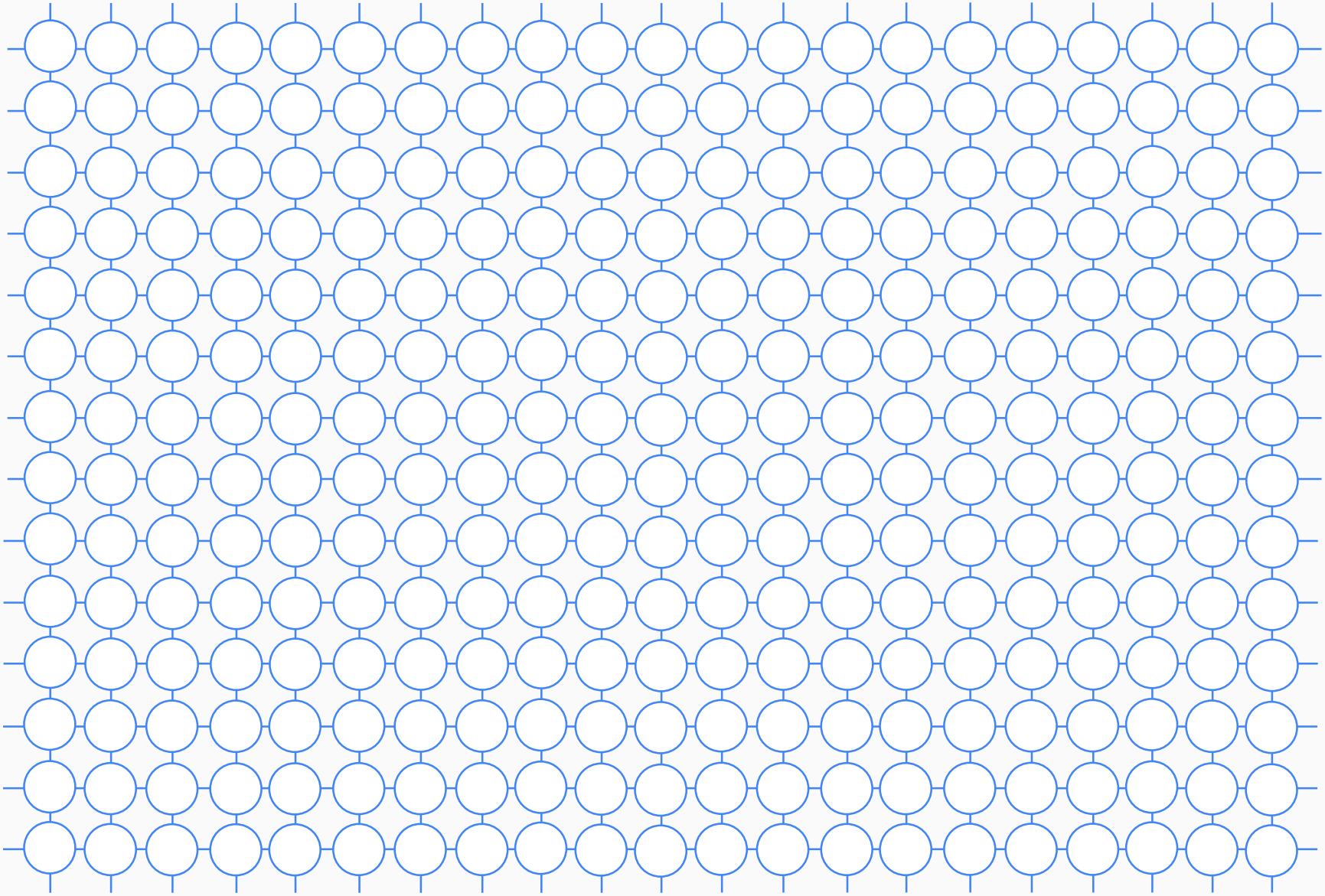
Màn hình CRT



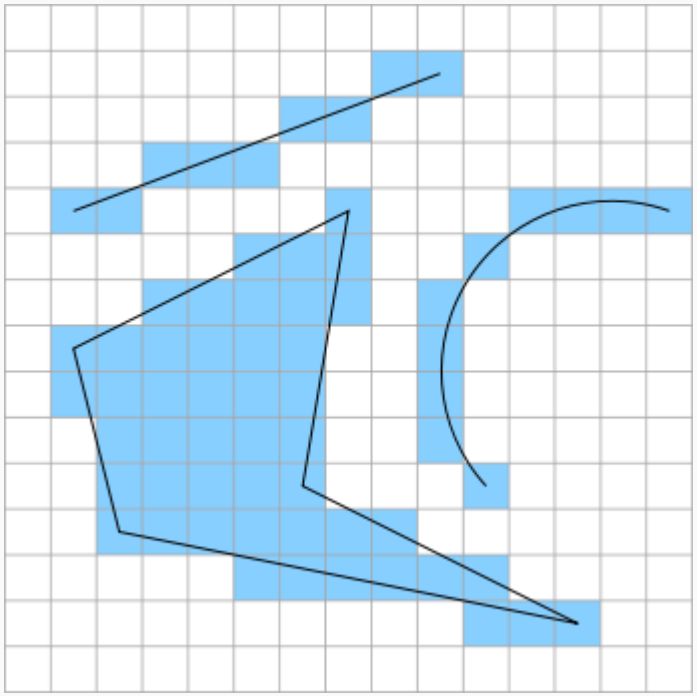


http://www.informatics.buzdo.com/p202-computer-monitor.htm

Lưới các điểm

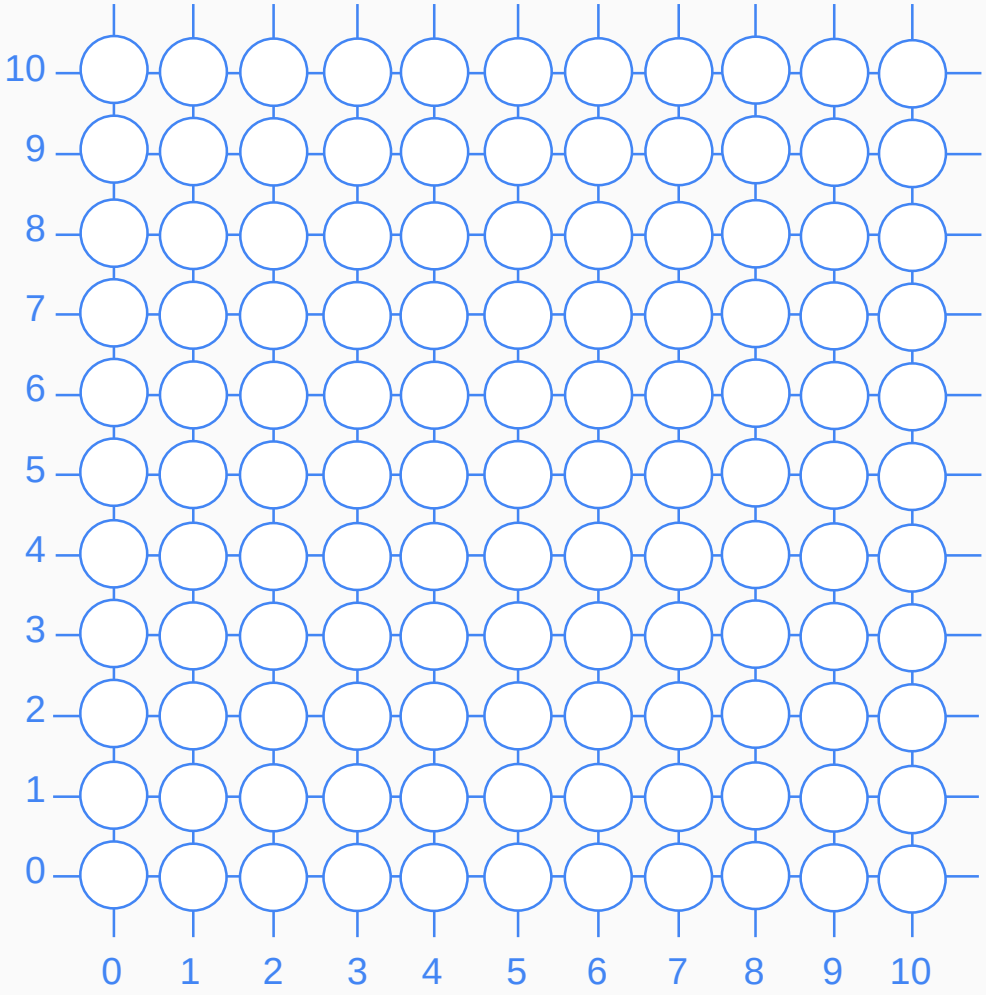


Quy trình hiển thị



- Miền giá trị của các điểm nằm trên đường cong thực là (x,y) $\in$ R.
- Miền giá trị của các điểm trên lưới pixel là các điểm (x,y) $\in$N
- Làm thế nào để xấp xỉ các điểm nguyên để tạo hình ảnh giống như đường cong thực ?

Quy trình hiển thị



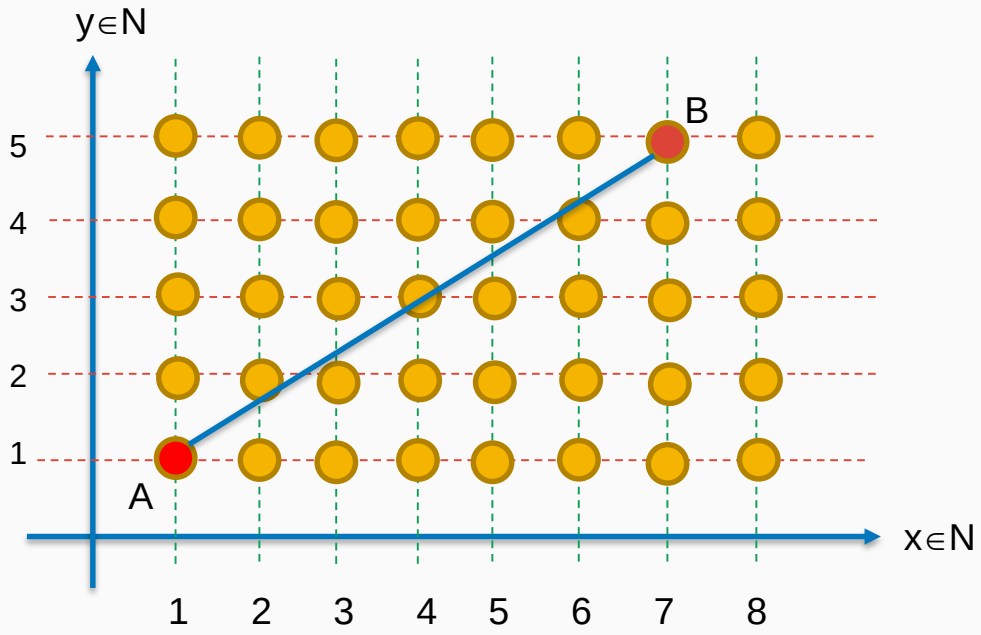
Các đường cơ bản

- Vẽ đoạn thẳng
  - Thuật toán DDA (Digital Differential Analyzer)
  - Thuật toán Bresenham
- Vẽ đường tròn
  - Thuật toán MidPoint
  - Thuật toán Bressenham
- Vẽ các đường cong

Vẽ đoạn thẳng

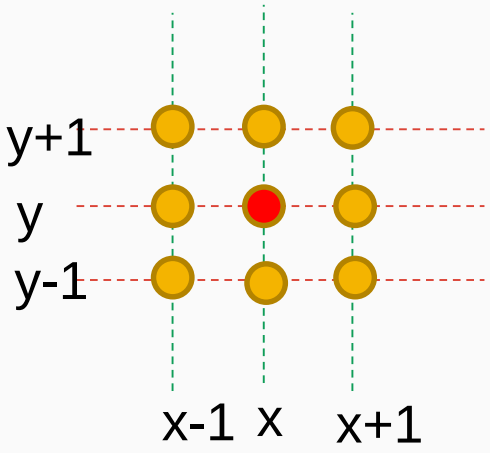
Vẽ đoạn thẳng

- Ví dụ 1: Xác định vị trí của các điểm sẽ được vẽ lên màn hình để tạo thành đường thẳng từ A(1,1) đến B(7,5)



Vẽ đoạn thẳng

- Các điểm thuộc vị trí lân cận 8 của điểm (x,y) sẽ tạo thành vệt sáng liên tục mà mắt thường khó phân biệt.



Các điểm thuộc lân cận 8 của (x,y)

Vẽ đoạn thẳng

- Xét phương trình đường thẳng qua 2 điểm A(1,1), B(7,5):

$$\frac{x-x_1}{x_2-x_1} = \frac{y-y_1}{y_2-y_1} \rightarrow y = \frac{y_2-y_1}{x_2-x_1}(x - x_1) + y_1 \ (1)$$

- Đặt: $Dx = x_2 - x_1, Dy = y_2 - y_1, m = \frac{Dy}{Dx}, b = y_1 - mx_1$
- Khi đó phương trình (1) có thể viết lại như sau:

$$y = mx + b \ (2)$$

- Cho giá trị x thay đổi từ x_1 đến x_2, thế x vào (2) ta tính được giá trị của y.

Vẽ đoạn thẳng

- Cụ thể: $Dx = 6, Dy = 4, m = \frac{4}{6} = \frac{2}{3} \sim 0.67, b = \frac{1}{3} \sim 0.33$
- Cho giá trị x thay đổi từ x_1=1 đến x_2=7, thế x vào (2) ta tính được giá trị của y.

x	1	2	3	4	5	6	7
y	1.00	1.67	2.34	3.01	3.68	4.35	5.02

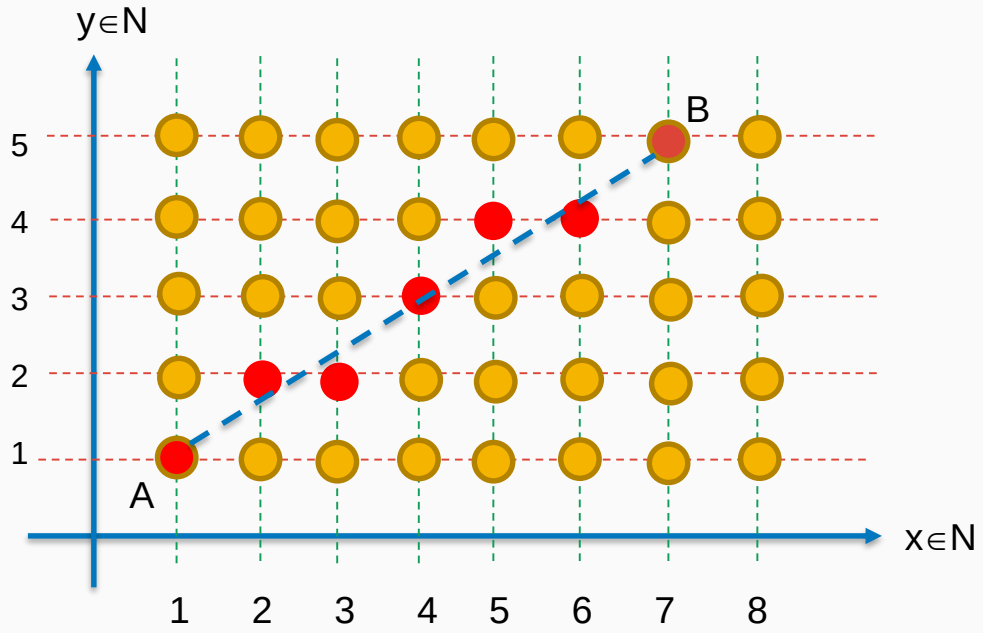
- Tuy nhiên, các giá trị y là số thực → cần làm tròn giá trị thành số nguyên

x	1	2	3	4	5	6	7
y	1.00	1.67	2.34	3.01	3.68	4.35	5.02
y làm tròn	1	2	2	3	4	4	5

Vẽ đoạn thẳng

- Ví dụ 1: Xác định vị trí của các điểm sẽ được vẽ lên màn hình để tạo thành đường thẳng từ A(1,1) đến B(7,5)

x	1	2	3	4	5	6	7
y	1.00	1.67	2.34	3.01	3.68	4.35	5.02
y làm tròn	1	2	2	3	4	4	5



Vẽ đoạn thẳng

- Nhận xét: Khi x thay đổi, ta tính y=m*x+b.

→ Giảm chi phí tính toán ???

- Giả sử tại $x = x_i$, ta tính được y $= y_i = m * x_i + b$

- Khi x tăng lên 1: $x = x_{i+1} = x_i + 1$, ta tính y như sau:

$$y = y_{i+1} = m * x_{i+1} + b = m * (x_i + 1) + b$$
$$= m * x_i + b + m = y_i + m$$

- → Từ 1 phép nhân và 1 phép cộng số thực $(m * x + b)$, ta chỉ cần 1 phép cộng số thực $(y_i + m)$ để tính y.

Vẽ đoạn thẳng

- Tóm lại: để xác định vị trí của các điểm sẽ được vẽ lên màn hình để tạo thành đường thẳng từ A(x1,y1) đến B(x2,y2):
  - Bước 1: Tính $Dx = x_2 - x_1, Dy = y_2 - y_1, m = \frac{Dy}{Dx}$
  - Bước 2: Khởi tạo tọa độ điểm đầu tiên: x=x1, y=y1
  - Bước 3: for x=x1+1 to x2 do

    y = y+m;

    y' = round(y)

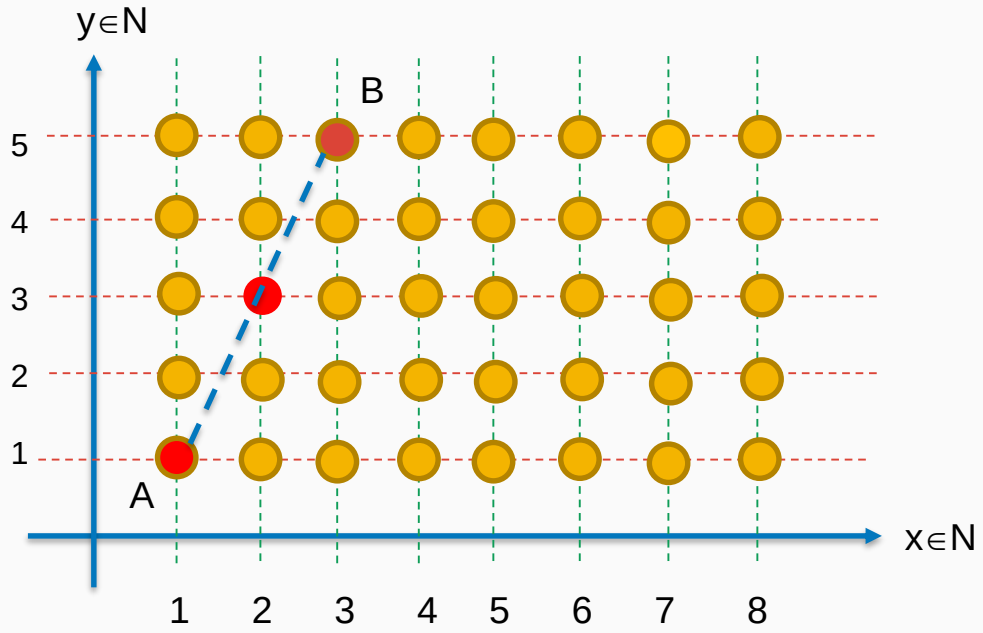
    Tọa độ cần xác định (x,y');

    end

Vẽ đoạn thẳng

- Ví dụ 2: Áp dụng phương pháp trên với A(1,1) đến B(3,5):
  - $Dx = x_2 - x_1 = 2, Dy = y_2 - y_1 = 4, m = \frac{Dy}{Dx} = 2$

x	1	2	3
y	1	3	5
y'= round(y)	1	3	5

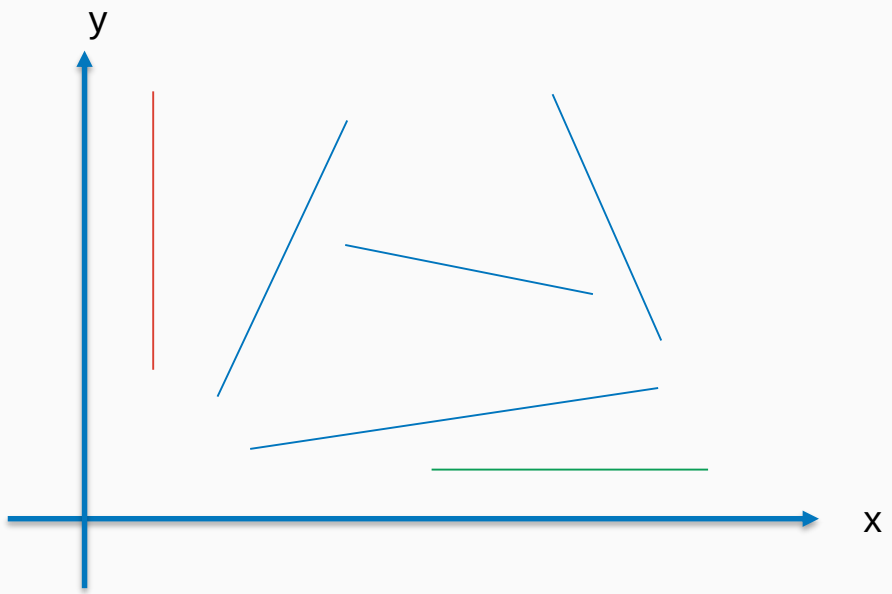


Đoạn thẳng AB không liền nét !!

Vẽ đoạn thẳng

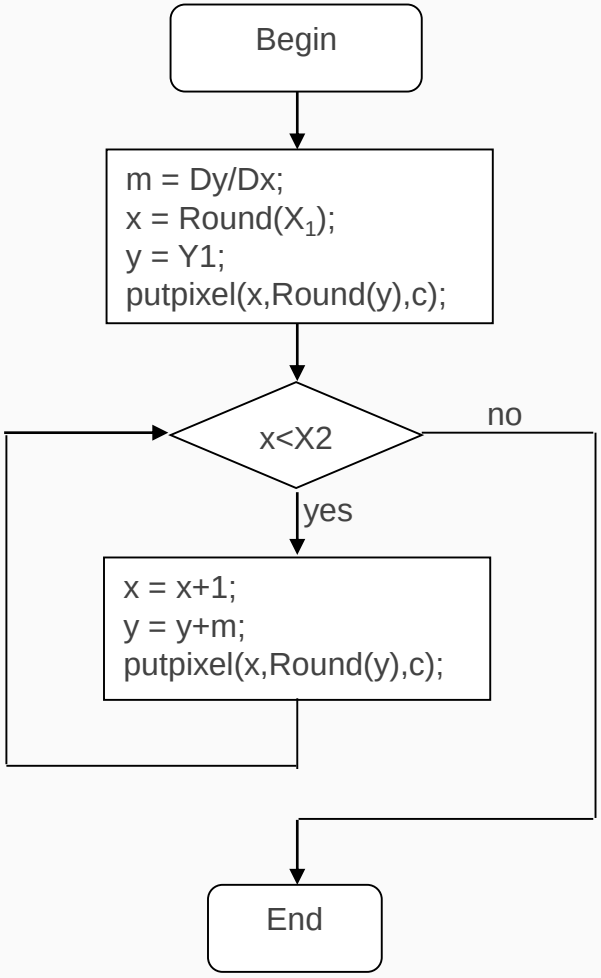
➔ Cần xác định các trường hợp khác nhau

- Có các trường hợp của hệ số góc m như sau:
  - Dx=0: đoạn thẳng AB song song với trục y
  - Dy=0: đoạn thẳng AB song song với trục x
  - 0<m<1
  - m>1
  - 0>m>-1
  - m< -1



Thuật toán DDA

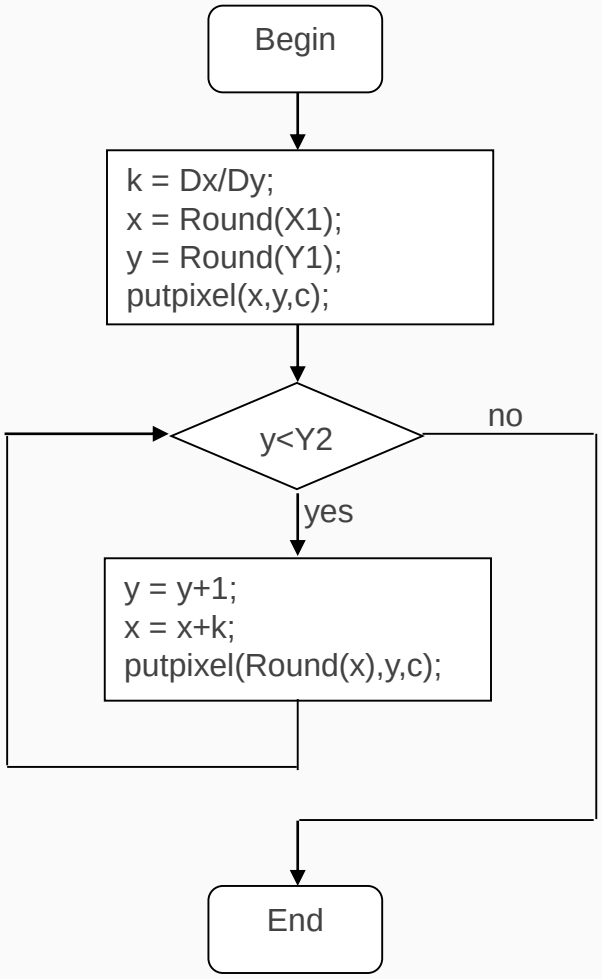
- Xét trường hợp 0<m<1



- Tham khảo thêm: https://en.wikipedia.org/wiki/Digital_differential_analyzer_(graphics_algorithm)

Thuật toán DDA

- Xét trường hợp m>1



Thuật toán DDA

- Nhận xét:
  - Giá trị y (hoặc x) được tính dựa trên phép cộng số thực (y=y+m (hoặc x=x+k))
  - Phải mất chi phí làm tròn giá trị thực thành giá trị nguyên (Round(float x) ~ int(x+0.5)).

- ➔ Cần được cải tiến để giảm chi phí tính toán.

Thuật toán Bresenham

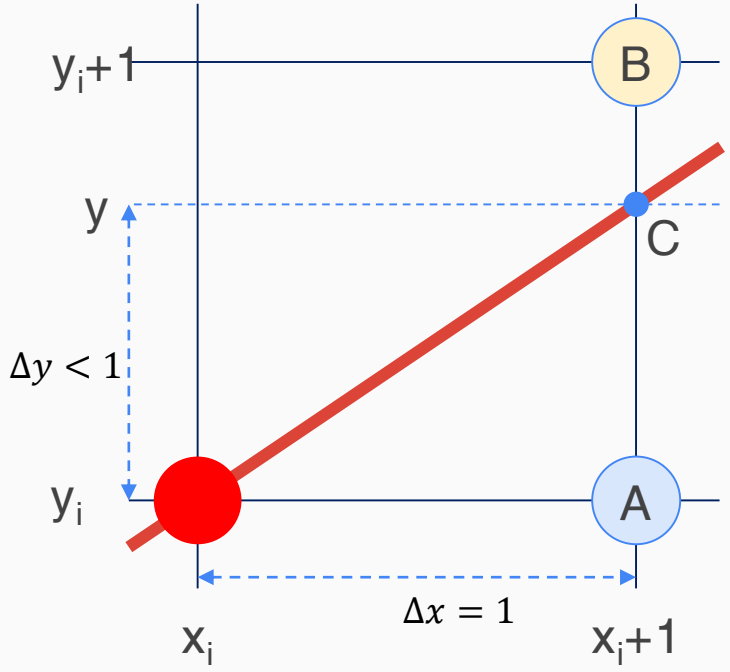
- The Bresenham algorithm is another incremental scan conversion algorithm
- The big advantage of this algorithm is that it uses only integer calculations

Jack Bresenham worked for 27 years at IBM before entering academia. Bresenham developed his famous algorithms at IBM in the early 1960s

Thuật toán Bresenham

Xét trường hợp 0<m<1

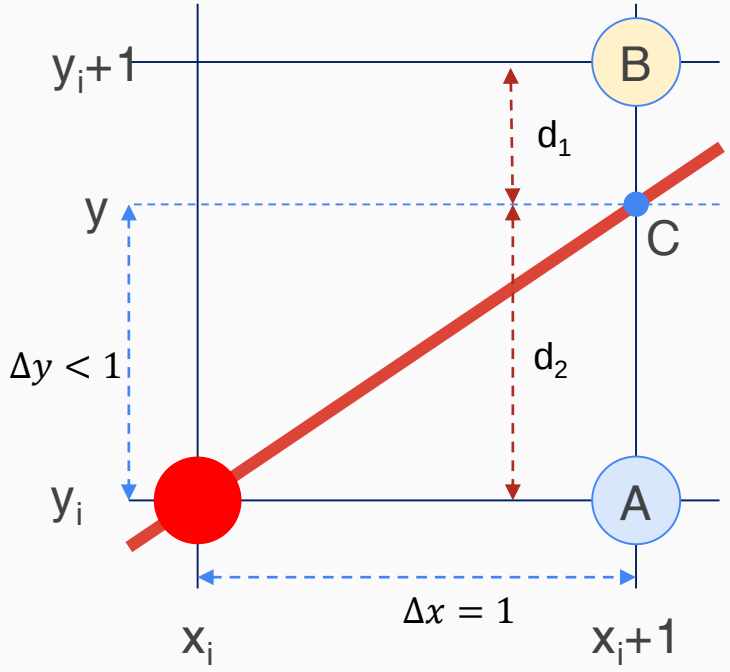
- Trong trường hợp này, $m < 1 \rightarrow |D_y| < |D_x|$
  - → độ biến đổi theo trục x nhiều hơn trục y → cho x thay đổi rồi tính y theo x.
  - → Khi x thay đổi 1 lượng $\Delta x = +1$ thì y thay đổi một lượng $\Delta y < 1$
  - Giả sử $D_x > 0$



Thuật toán Bresenham: 0<m<1

- Gọi C là điểm thuộc đường thẳng gốc tại $x=x_i+1$.
- Gọi $d_1=BC = (y_i+1 – y)$ và $d_2=CA = (y-y_i)$
  - Nếu d2>d1 → điểm C gần điểm B hơn → điểm B là điểm sẽ được chọn để vẽ.
  - Ngược lại → điểm C gần điểm A hơn → điểm A là điểm sẽ được chọn để vẽ.

- Tuy nhiên, để tính d_1,d_2 ta phải tính y dựa vào phương trình đường thẳng qua A, B → tính trên số thực.
- → Khắc phục bằng cách xét dấu d_2-d_1

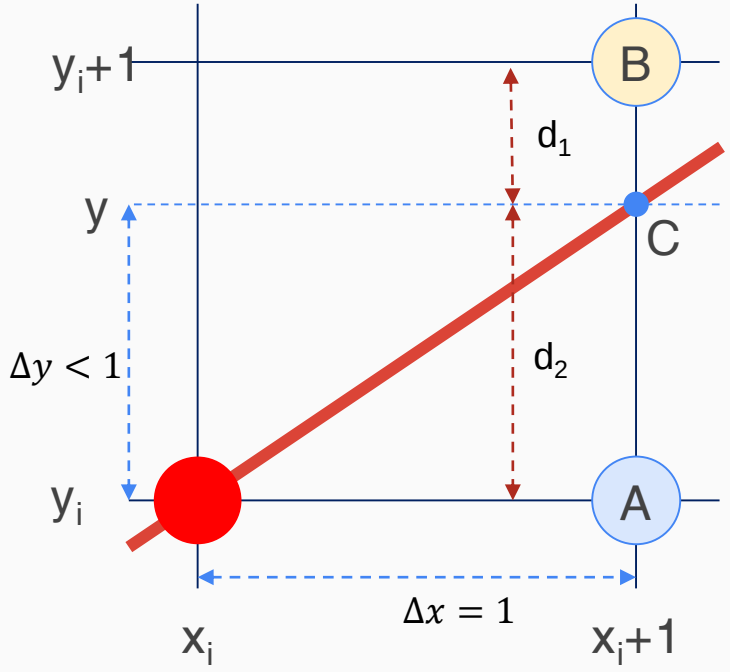


Thuật toán Bresenham: 0<m<1

- Đặt p = d_2-d_1= (y-y_i) – (y_i+1-y) = 2y-2y_i-1

$$p = 2(mx + b) - 2y_i - 1 = 2\left(\frac{Dy}{Dx}(x_i + 1) + (\frac{Dy}{Dx}x_1 - y_1)\right) - 2y_i - 1$$

- Do giá trị của p phụ thuộc vào (xi,yi) nên ta đặt p là pi.
- Ngoài ra, trong biểu thức của p có Dx ở dưới mẫu → p là số thực → khử Dx bằng cách xét p=D_x(d_2-d_1).
- Do Dx>0 nên dấu của p=(d2-d1) và dấu của p=D_x(d_2-d_1) như nhau.



Thuật toán Bresenham: 0<m<1

- Khi đó ta có:

$$p_i = 2D_y x_i - 2D_x y_i + 2D_y - D_x - 2D_y x_1 + 2D_x y_1$$

- Xét:

$$p_{i+1} - p_i = 2D_y(x_{i+1} - x_i) - 2D_x(y_{i+1} - y_i)$$

- Nếu $p_i > 0 \leftrightarrow d_2 > d_1$, điểm vẽ được chọn là B:

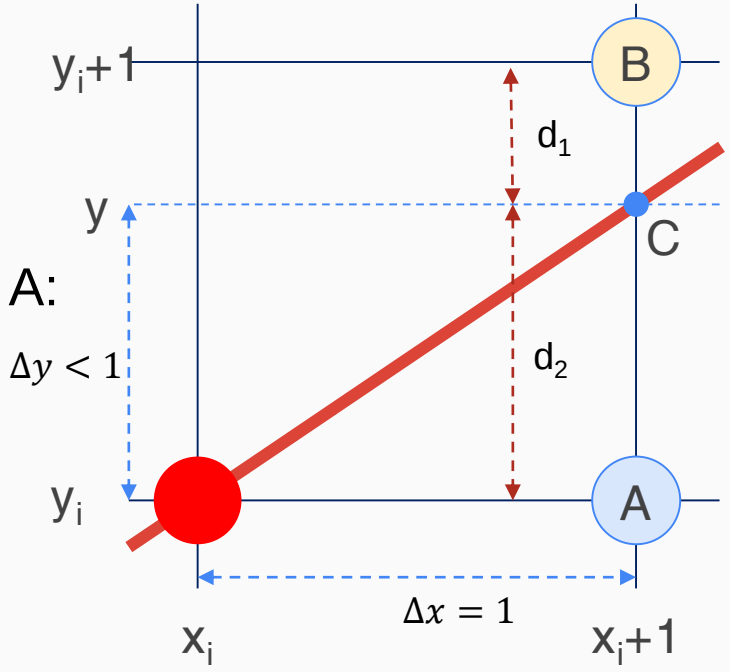
$$\begin{cases} x_{i+1} = x_i + 1 \\ y_{i+1} = y_i + 1 \end{cases}$$

Khi đó: $p_{i+1} = p_i + 2D_y - 2D_x$

- Ngược lại, điểm vẽ được chọn là A:

$$\begin{cases} x_{i+1} = x_i + 1 \\ y_{i+1} = y_i \end{cases}$$

Khi đó: $p_{i+1} = p_i + 2D_y$



Thuật toán Bresenham: 0<m<1

- Tính giá trị p_1 đầu tiên

$$p_1 = 2D_y x_1 - 2D_x y_1 + 2D_y - D_x - 2D_y x_1 + 2D_x y_1$$
$$= 2D_y - D_x$$

- Tóm tắt thuật toán trường hợp 0<m<1
- Bước 1: Khởi tạo

$$D_x = x_2 - x_1, D_y = y_2 - y_1$$
$$\mathrm{p} = 2D_y - D_x$$
$$x = x_1$$
$$y = y_1$$

- Bước 2: Trong khi $x < x_2$
  - Nếu $p > 0$, tọa độ đỉnh tiếp theo và giá trị p tiếp theo như sau:

$$\begin{cases} x_{i+1} = x_i + 1 \\ y_{i+1} = y_i + 1 \\ p = p + 2D_y - 2D_x \end{cases}$$

  - Ngược lại, ta có:

$$\begin{cases} x_{i+1} = x_i + 1 \\ y_{i+1} = y_i \\ p = \mathrm{p} + 2D_y \end{cases}$$

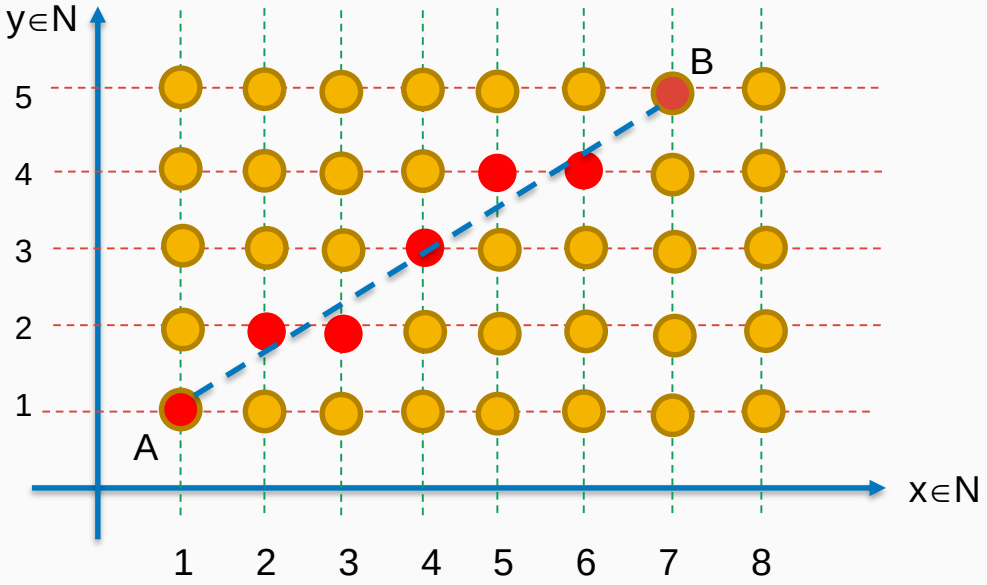
Vẽ đoạn thẳng

- Xét lại ví dụ 1: Xác định vị trí của các điểm sẽ được vẽ lên màn hình để tạo thành đường thẳng từ A(1,1) đến B(7,5)

$D_x = x_2 - x_1 = 6, D_y = y_2 - y_1 = 4,\ \mathrm{p} = 2D_y - D_x = 2$
$x = x_1 = 1; y = y_1 = 1$

p	2	-2	6	2	-2	6	2
x	1	2	3	4	5	6	7
y	1	2	2	3	4	4	5

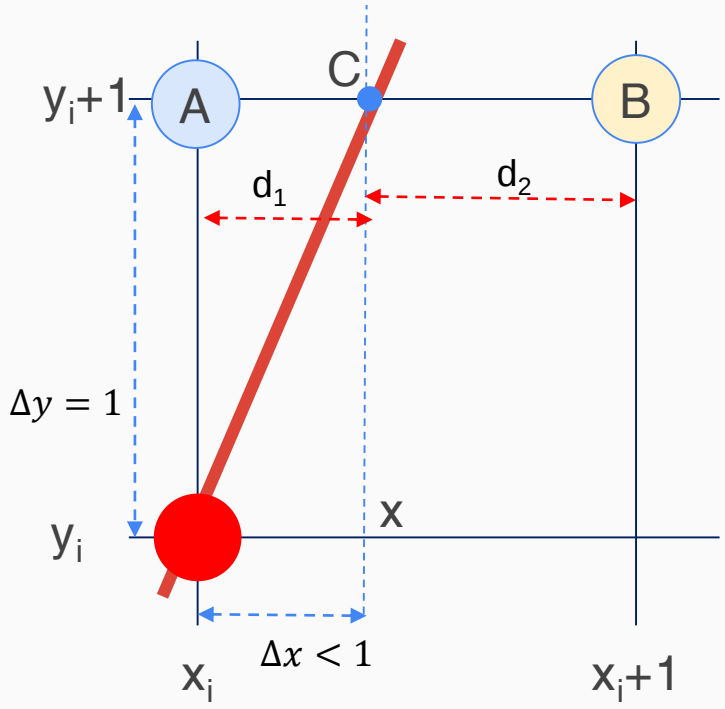
$2D_y - 2D_x = -4$
$2D_y = 8$



Thuật toán Bresenham

Xét trường hợp m>1

- Trong trường hợp này, $m > 1 \rightarrow |D_y| > |D_x|$
  - → độ biến đổi theo trục y nhiều hơn trục x → cho y thay đổi rồi tính y theo x.
  - → Khi y thay đổi 1 lượng $\Delta y = +1$ thì x thay đổi một lượng $\Delta x < 1$
  - Giả sử $D_y > 0$



Thuật toán Bresenham: m>1

- Xét tương tự như trường hợp 0<m<1, ta sẽ có thuật toán như sau:
- Bước 1: Khởi tạo

$$D_x = x_2 - x_1, D_y = y_2 - y_1$$
$$\mathrm{p} = 2D_x - D_y$$
$$x = x_1$$
$$y = y_1$$

- Bước 2: Trong khi $y < y_2$
  - Nếu $p > 0$, tọa độ đỉnh tiếp theo và giá trị p tiếp theo như sau:

$$\begin{cases} x_{i+1} = x_i + 1 \\ y_{i+1} = y_i + 1 \\ p = p + 2D_x - 2D_y \end{cases}$$

  - Ngược lại, ta có:

$$\begin{cases} x_{i+1} = x_i \\ y_{i+1} = y_i + 1 \\ p = \mathrm{p} + 2D_x \end{cases}$$

Thuật toán Bresenham

- Nhận xét:
  - Thuật toán chỉ tính toán trên số nguyên.
  - Việc xác định vị trí (tọa độ) các điểm chỉ phụ thuộc vào dấu của p
  - Dựa vào dấu của p để xác định tọa độ và giá trị p kế tiếp.

Vẽ đường tròn

Circle Boundary

Vẽ đường tròn

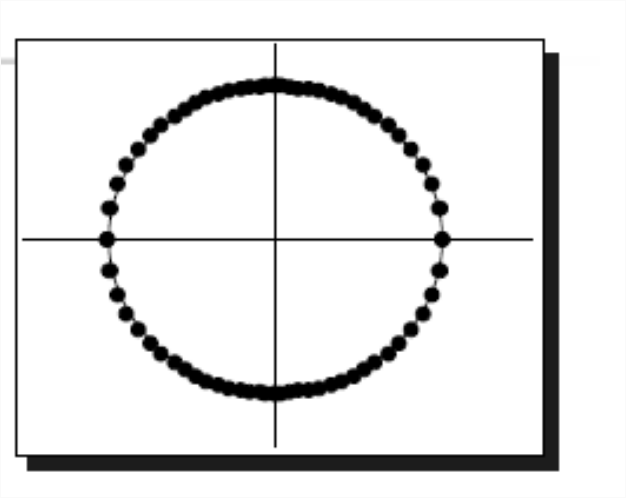
1. Using Explicit Representation
2. Using Parametric Representation
3. Midpoint Circle Drawing Algorithm

1-Drawing Circles Using Explicit Representation

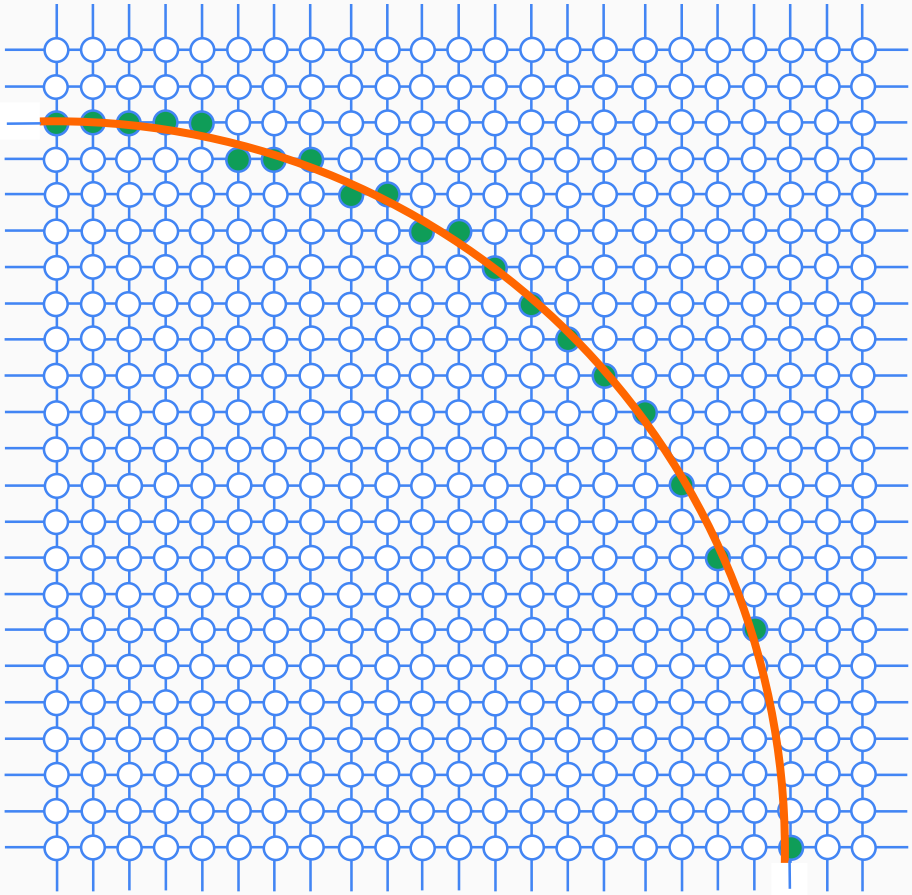
$$(x - xc)^2 + (y - yc)^2 = r^2$$

$$y = yc \pm \sqrt{r^2 - (x - xc)^2}$$

- Very simple
- Considerable computation at each step.
- The spacing between plotted pixel positions is not uniform.



1-Drawing Circles Using Explicit Representation



$$y_0 = \sqrt{20^2 - 0^2} \approx 20$$

$$y_1 = \sqrt{20^2 - 1^2} \approx 20$$

$$y_2 = \sqrt{20^2 - 2^2} \approx 20$$

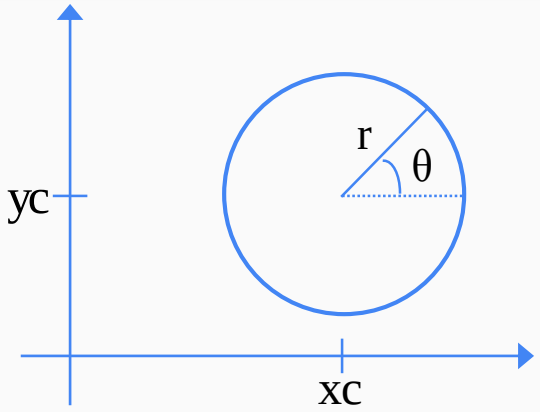
$$\vdots$$

$$y_{19} = \sqrt{20^2 - 19^2} \approx 6$$

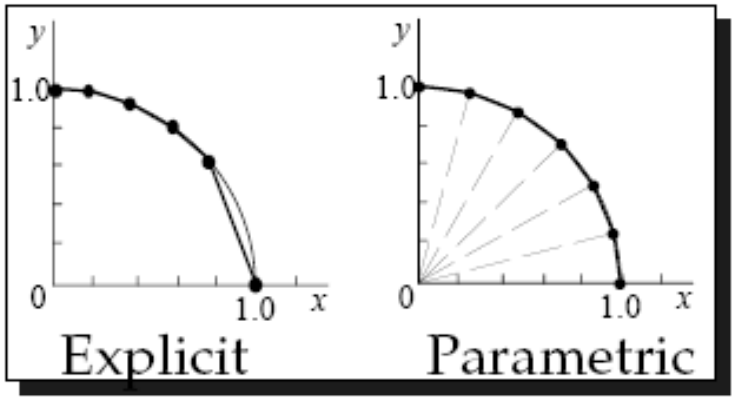
$$y_{20} = \sqrt{20^2 - 20^2} \approx 0$$

2- Drawing Circles Using Parametric Representation

$$x = xc + r\cos\theta$$
$$y = yc + r\sin\theta$$



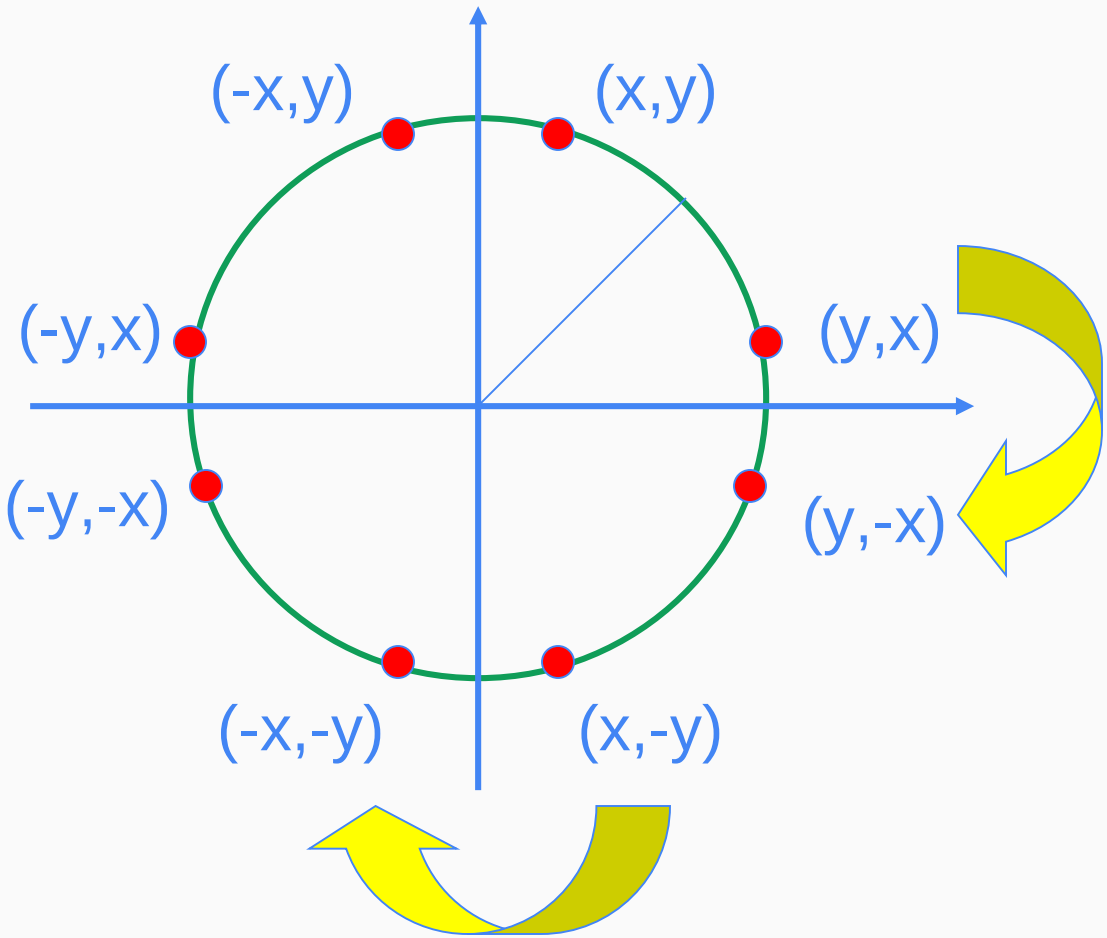
$$x_{i+1} = x_i \cos\delta\theta - y_i \sin\delta\theta$$
$$y_{i+1} = x_i \sin\delta\theta + y_i \cos\delta\theta$$



Still considerable computation at each step.

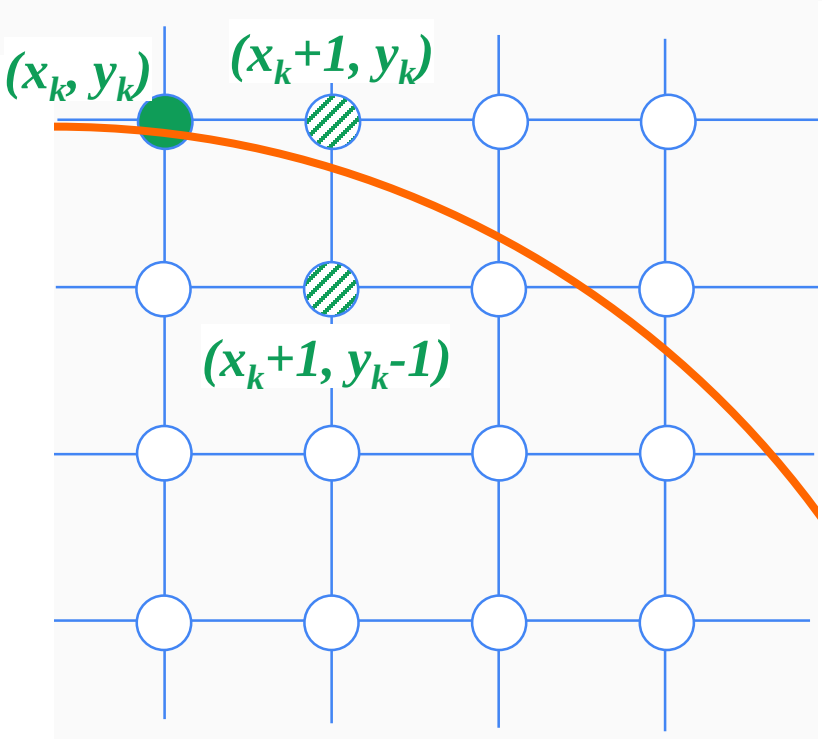
Optimisation and speed-up

- Symmetry of a circle can be used
- Calculations of point coordinates only for a first one-eighth of a circle



Thuật toán Mid-Point (Bresenham)

- Assume that we have just plotted point (x_k, y_k)
- The next point is a choice between (x_k+1, y_k) and (x_k+1, y_k-1)
- We would like to choose the point that is nearest to the actual circle
- So how do we make this choice?



Thuật toán Mid-Point (Bresenham)

- The equation of the circle slightly to give us:

$$f_{circ}(x, y) = x^2 + y^2 - r^2$$

- The equation evaluates as follows:

$$f_{circ}(x, y) \begin{cases} < 0, \text{ if } (x, y) \text{ is inside the circle boundary} \\ = 0, \text{ if } (x, y) \text{ is on the circle boundary} \\ > 0, \text{ if } (x, y) \text{ is outside the circle boundary} \end{cases}$$

- By evaluating this function at the midpoint between the candidate pixels we can make our decision

Thuật toán Mid-Point (Bresenham)

- Assuming we have just plotted the pixel at (x_k,y_k) so we need to choose between (x_k+1,y_k) and (x_k+1,y_k-1)
- Our decision variable can be defined as:

$$p_k = f_{circ}(x_k+1, y_k - \tfrac{1}{2})$$

$$= (x_k+1)^2 + (y_k - \tfrac{1}{2})^2 - r^2$$

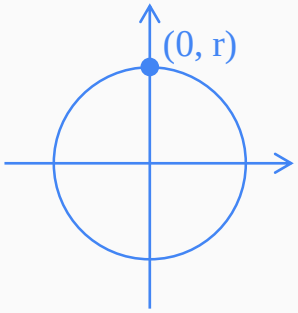
- If $p_k < 0$ the midpoint is inside the circle and the pixel at y_k is closer to the circle
- Otherwise the midpoint is outside and y_k-1 is closer

Thuật toán Mid-Point (Bresenham)

$$let \quad p_k = f(x_k + 1, y_k - \frac{1}{2}) = (x_k + 1)^2 + (y_k - \frac{1}{2})^2 - r^2$$

$$p_{k+1} = f(x_{k+1} + 1, y_{k+1} - \frac{1}{2}) = (x_k + 2)^2 + (y_{k+1} - \frac{1}{2})^2 - r^2$$

$$where \quad x_{k+1} = x_k + 1$$



$$y_{k+1} = \begin{cases} y_k & if \quad p_k < 0 \\ y_k - 1 & \text{otherwise} \end{cases}$$

$$p_{k+1} = \begin{cases} p_k + 2x_k + 3 & if \quad p_k < 0 \\ p_k + 2(x_k - y_k) + 5 & \text{otherwise} \end{cases}$$

$$p_0 = f(x_0 + 1, y_0 - \frac{1}{2}) = (0+1)^2 + (r - \frac{1}{2})^2 - r^2 = \frac{5}{4} - r$$

since $(x_0, y_0) = (0, r)$

Thuật toán Mid-Point (Bresenham)

MID-POINT CIRCLE ALGORITHM

1. Input radius r and circle centre (x_c, y_c), then set the coordinates for the first point on the circumference of a circle centred on the origin as:

$$(x_0, y_0) = (0, r)$$

2. Calculate the initial value of the decision parameter as:

$$p_0 = 5/4 - r \cong 1 - r$$

3. Starting with $k = 0$ at each position x_k, perform the following test. If $p_k < 0$, the next point along the circle centred on $(0, 0)$ is (x_k+1, y_k) and:

$$p_{k+1} = p_k + 2x_k + 3$$

Otherwise the next point along the circle is (x_k+1, y_k-1) and:

$$p_{k+1} = p_k + 2(x_k - y_k) + 5$$

Thuật toán Mid-Point (Bresenham)

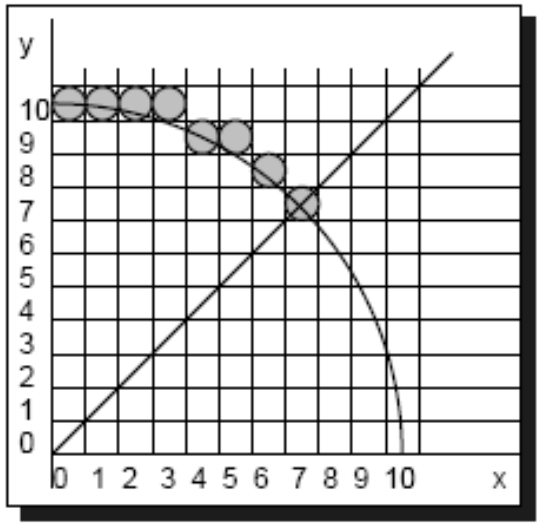
4. Determine symmetry points in the other seven octants
5. Move each calculated pixel position (x, y) onto the circular path centred at (x_c, y_c) to plot the coordinate values:

$$x = x + x_c \qquad y = y + y_c$$

6. Repeat steps 3 to 5 until $x >= y$

Midpoint Circle Algorithm. Example

r=10,
p_0=1-r=-9,
(x_0,y_0)= (0,10),
$2x_0$=0, $2y_0$=20



k	p_k	(x_{k+1},y_{k+1})	$2x_{k+1}$	$2y_{k+1}$
0	-9	(1,10)	2	20
1	-6	(2,10)	4	20
2	-1	(3,10)	6	20
3	6	(4,9)	8	18
4	-3	(5,9)	10	18
5	8	(6,8)	12	16
6	5	(7,7)	14	14

Tài liệu tham khảo

- Slide này được biên soạn được tham khảo từ một số tài liệu sau:
  - https://en.wikipedia.org/wiki/Bresenham's_line_algorithm
  - http://jcsites.juniata.edu/faculty/rhodes/graphics/verts2frags2.htm
  - Slide: Computer Graphics 4: Bresenham Line Drawing Algorithm, Circle Drawing & Polygon Filling By:Kanwarjeet Singh