بيانيات حاسوبية

المحاضرة الثانية

د. غيث ابراهيم بلال

❖ خوارزمية المحلل التفاضلي الرقمي DDA:

اختصار ل (Digital Differential Analyzer).

هي خوارزمية خاصة بالمسح الخطي وتتميز بإجراء حسابات عند كل خطوة باستخدام نتائج الخطوة السابقة.

وعلى افتراض أننا حسبنا الخطوة i أي النقطة أو البكسل (x_i , y_i) وبما أن النقطة الجديدة هي (x_{i+1} , y_{i+1}) يجب أن تحقق:

$$\Delta y = y_{i+1} - y_i \qquad \frac{\Delta y}{\Delta x} = m$$

$$\Delta x = x_{i+1} - x_i$$

$$* \quad \boxed{\begin{aligned} y_{i+1} &= y_i + m\,\Delta x \\ x_{i+1} &= x_i + \Delta y / m \end{aligned}}$$

وهنا نتابع بأخذ العينات من المستقيم عند وحدات متساوية مأخوذة باتجاه أحد

محوري الإحداثيات ويتم هنا تقرير القيم الصحيحة الموافقة والأقرب للخط من أجل المحور الإحداثي الآخر.

أي في حال كون الميل:

$$1.\quad |m|\le 1 : \Delta x = 1 \longrightarrow y_{i+1} = y_i + m$$

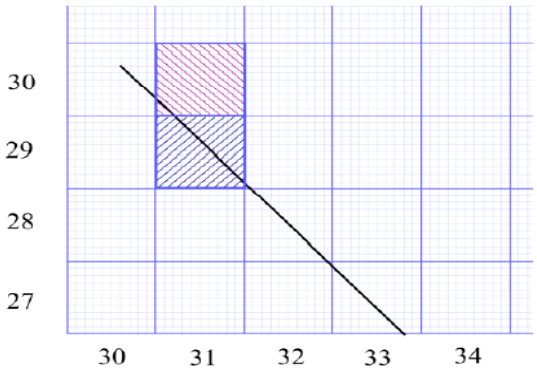
$$2.\quad |m|>1 : \Delta y = 1 \longrightarrow x_{i+1} = x_i + 1/m$$

هنا نأخذ القيم الصحيحة فقط، أي نطبق التحويل Round للقيم التي تحتوي على الميل (تقريب القيم إلى أقرب عدد صحيح)

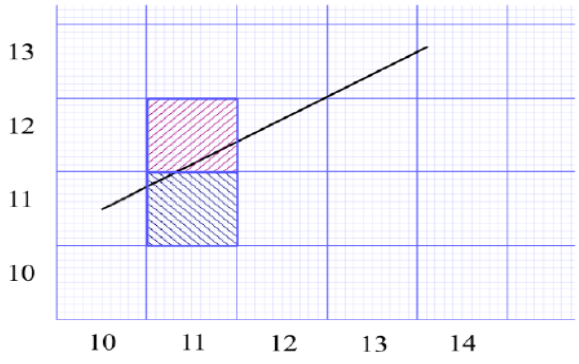
أي نأخذ القيم من 0 إلى النقطة الأخيرة. $i = [x_0 \rightarrow x_n]$

❖ **خوارزمية بريزنهام لمسح المستقيم (Bresenham's Line Algorithm):**

تعمل خوارزمية بريزنهام وفق المبدأ التالي:



B



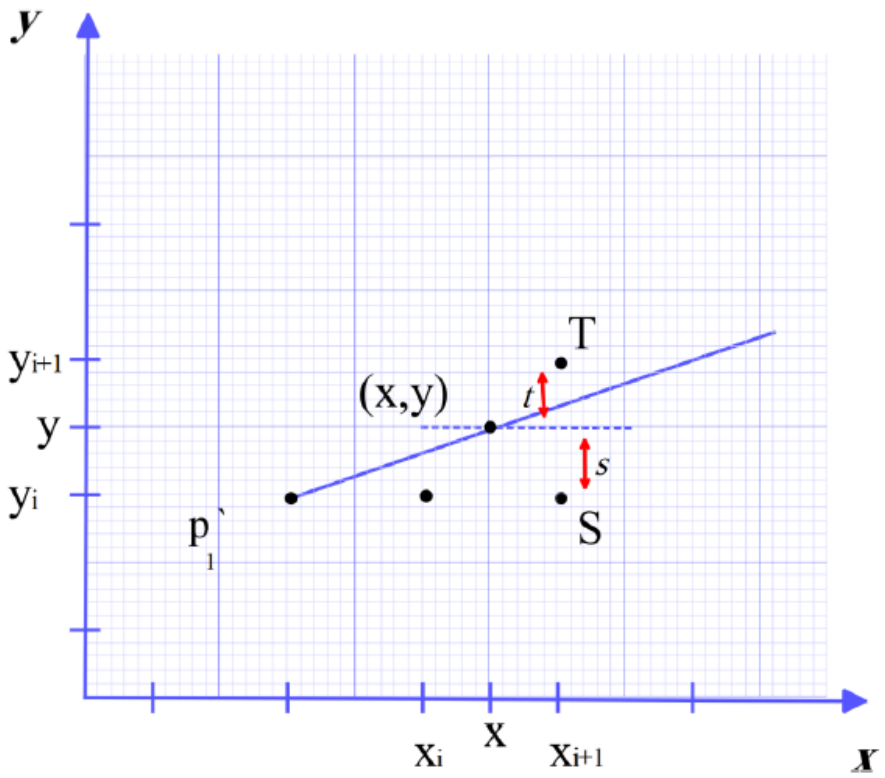
A

هنا نبدأ بالطرق الأيسر بالنسبة إلى البكسلات، ويجب أن نقرر ما هو الموقع التالي

للبكسلات، هل هو (11,11) أم (11,12) بالنسبة إلى الشكل A.

والشكل B: يجب أن نقرر الموقع التالي للبكسل هل هو (31,30) أم (31,29)

- تبدأ هذه الخوارزمية بالبكسل الأول $p_1`(x_1`,y_1`)$ ثم نأخذ البكسلات التالية باتجاه اليمين باختيار بكسل واحد في كل مرة في الاتجاه الأفقي نحو آخر بكسل على فرض أنه $p_2`(x_2`,y_2`)$ على اعتبار أن $0<m<1$



بفرض أن آخر بكسل تم اختياره هو(x_i , y_i) ومهمتنا الآن أن نقرر اختيار أي بكسل، السفلي S أم العلوي T.

- إذا اخترنا S السفلي نجد أن: $x_{i+1} = x_{i+1}$ & $y_{i+1} = y_i$
- إذا اخترنا T العلوي نجد أن: $x_{i+1} = x_{i+1}$ & $y_{i+1} = y_{i+1}$

وهي إحداثيات البكسل المراد اختياره، وبالتالي النقطة y التي تحقق معادلة المستقيم

لأجل $x = x_{i+1}$ هي $y = m\,x_{i+1} + b$

نبدل كل $x_{i+1} = x_i + 1$ ⟵ $\boxed{y = m(x_i+1) + b}$ *

- المسافة s تساوي : $s = y - y_i$
- والمسافة t تساوي : $t = (y_i + 1) - y$ على اعتبار أن $y_{i+1} = y_i + 1$

الآن ندرس الفرق بين المسافتين لكي نقرر أي بكسل نأخذ:

1. إذا كان $s - t < 0$: أي $s < t$ يكون البكسل الأقرب هو S.
2. إذا كان $s - t > 0$: أي $s > t$ يكون البكسل الأقرب هو T.
3. إذا كان $s - t = 0$: هنا المسافة واحدة، لذلك نختار أي منهما.

- وبالتالي ندرس الفرق:

$$s - t = (y - y_i) - [(y_i + 1) - y] \longrightarrow s - t = 2y - 2y_i - 1$$

ومن المعادلة * نعوض Y بقيمتها:

$$s - t = 2m(x_i + 1) + 2b - 2y_i - 1$$

الآن لدينا الميل $m = \frac{\Delta y}{\Delta x}$ وندخل هنا معامل جديد نسميه معامل القرار أو

(متغير القرار) ونرمز له d ويساوي $d_i = \Delta x\,(s - t)$ ، نعوض بالمعادلة:

$$\Delta x\,(s - t) = 2\Delta y\, x_i + 2\Delta y + 2b\,\Delta x - 2y_i\,\Delta x - \Delta x$$

$$d_i = 2\Delta y\, x_i - 2y_i\,\Delta x + c$$

حيث أن $c = 2\Delta y + \Delta x\,(2b - 1)$

وبالتالي معامل القرار (متغير القرار) للنقطة الجديدة هو:

$$d_{i+1} = 2\Delta y\, x_{i+1} - 2y_{i+1}\,\Delta x + c$$

نأخذ الفرق بين العاملين:

$$d_{i+1} - d_i = 2\Delta y\,(x_{i+1} - x_i) - 2\Delta x\,(y_{i+1} - y_i)$$

وبما أن $x_{i+1} = x_i + 1$ وبالتالي:

** $\boxed{d_{i+1} = d_i + 2\Delta y - 2\Delta x\,(y_{i+1} - y_i)}$

1. إذا كان البكسل المأخوذ هو البكسل العلوي T:

أي $(d_i \geq 0)$ فإن $y_{i+1} = y_i + 1$ ⟵ نبدل في **

$$d_{i+1} = d_i + \underbrace{2(\Delta y - \Delta x)}$$

مقدار الزيادة في كل خطوة

2. إذا كان البكسل المأخوذ هو S (أي $d_i < 0$) فإن $y_{i+1} = y_i$ وبالتالي:

$$d_{i+1} = d_i + \underbrace{2\Delta y}$$

مقدار الزيادة

إذن:

معامل القرار للنقطة التالية

$$d_{i+1}=\begin{cases} d_i+2(\Delta y-\Delta x) & \text{if } d_i\geq 0 \\ d_i+2\Delta y & \text{if } d_i<0 \end{cases}$$

لكن يجب أن نحسب d_1 وهي قيمة حالة الأساس لهذه الصيغة التكرارية من الصيغة الأساسية لمتغير القرار (معامل القرار d)

$$d_1=\Delta x\,(s-t)=\Delta x\,[2m(x_1+1)+2b-2y_1-1]$$

$$d_1=\Delta x\,[2(m\,x_1+b-y_1)+2m-1]$$

وبما أن $m\,x_1+b-y_1=0$ دوماً

$$d_1=\Delta x\,[2m-1]=2m\Delta x-\Delta x=2\frac{\Delta y}{\Delta x}\Delta x-\Delta x$$

$$\boxed{d_1=2\Delta y-\Delta x}$$

نحسبها لأجل أول نقطة ثم نقرر أي بكسل سنأخذ S أم T وبالتالي يمكن كتابة خوارزمية بريزنهام لتحويل مسح مستقيم من $p`_1(x_1`, y_1`)$ إلى $p`_2(x_2`, y_2`)$ حيث أن $x`_1 < x`_2$ و $0<m<1$ كما يلي:

1. ننطلق من النقطتين (البداية والنهاية) للمستقيم $p`_1(x_1`, y_1`)$ والمستقيم $p`_2(x_2`, y_2`)$.
2. نأخذ النقطة الأولى $p`_1(x_1`, y_1`)$ ونمثل البكسل الأول.
3. نحسب كل من: Δx و Δy و $2\Delta y$ و $(2\Delta y-2\Delta x)$ ثم القيمة الابتدائية لمعامل القرار $d_o=2\Delta y-\Delta x$
4. من أجل قيمة ل x_k على طول الخط نقوم بما يلي $k = 1$:

a. إذا كانت $d_k < 0$ فإن النقطة (البكسل) التالية هي:

(x_k+1, y_k) و $d_{k+1} = d_k + 2\Delta y$

b. وإلا فإن البكسل الجديد هو (x_{k+1}, y_{k+1}) و

$$d_{k+1} = d_k + 2\Delta y - 2\Delta x$$

5. نكرر الخطوة الرابعة عدد من المرات يساوي $x_1 < x_2$.

ملحوظة:

إذا كان $x_1 > x_2$ للمستقيم المطلوب مسحه فإن الخطوة الرابعة تصبح كما يلي:

من أجل كل قيمة ل $(i = 1, 2, ...)$; y_i على طول الخط المستقيم نقوم بما

يلي:

1. إذا كانت $d_i < 0$ فإن البكسل التالي هو $(x_i - 1, y_i)$ و

$$d_{i+1} = d_i + 2\Delta y$$

2. و إلا فإن البكسل الجديد هو: $(x_i - 1, y_i + 1)$ و

$$d_{i+1} = d_i + 2\Delta y - 2\Delta x$$

3. نكرر هذه الخطوة طالما $y_i < y_2$

مثال1:

ارسم الخط المستقيم المحدد بالنقطتين (20,10) و (30,18).

1. $d_y = y_2 - y_1 = 18 - 10 = 8$
2. $\Delta x = x_2 - x_1 = 30 - 20 = 10$
3. $s = 2 * d_y = 16$

4. $t = 2*(d_y - d_x) = -4$

5. $d = 2d_y - d_x = 6$

نرسم أول بكسل وهو $(20,10)$

```
#include <iostream>
using namespace std;
void d_line(int x1, int y1, int x2, int y2)
{
    int dx = x2 - x1; int dy = y2 - y1;
    int s = 2 * dy;
    int t = 2 * (dy - dx);
    int d = 2 * dy - dx;
    cout << "d= " << d;
    int x = x1; int y = y1;
    cout << "\t pixel= " << x << "," << y << endl;
    while (x < x2) {
        cout << "d= " << d;
        if (d < 0)
        {
            d += s; x++;
        }
        else { d += t; y++; x++; }
        cout << "\t pixel= " << x << "," << y << endl;
    }
}
    void main() {d_line(20, 10, 30, 18);}
```

```
d= 6      pixel= 20,10
d= 6      pixel= 21,11
d= 2      pixel= 22,12
d= -2     pixel= 23,12
d= 14     pixel= 24,13
d= 10     pixel= 25,14
d= 6      pixel= 26,15
d= 2      pixel= 27,16
d= -2     pixel= 28,16
d= 14     pixel= 29,17
d= 10     pixel= 30,18
```

مثال2: طبق خوارزمية بريزنهام لمسح المستقيم المحدد بالنقطتين $p_1 = (1,1)$ و $p_2^{`} = (8,5)$:

الناتج هو كالتالي:

نحسب d_1 ، ثم نأخذ أول نقطة ك بيكسل مختار ثم نقرر بناءً على d_1 هل نأخذ البكسل الأعلى أم الأسفل.

```cpp
#include <iostream>
using namespace std;
void d_line(int x1, int y1, int x2, int y2)
{
    int dx = x2 - x1;  int dy = y2 - y1;
    int s = 2 * dy;
    int t = 2 * (dy - dx);
    int d = 2 * dy - dx;  cout << "d= " << d;
    int x = x1; int y = y1;
    cout << "\t pixel = " << x << "," << y << endl;
    while (x < x2) {
        cout << "d= "<< d;
        if (d < 0)
        {
            d += s;  x++;
        }
        else { d += t; y++; x++; }
        cout << "\t pixel = " << x << "," << y << endl; } }

void main()
{
    d_line(1, 1, 8, 5);
}
```

```
d= 1     pixel = 1,1
d= 1     pixel = 2,2
d= -5    pixel = 3,2
d= 3     pixel = 4,3
d= -3    pixel = 5,3
d= 5     pixel = 6,4
d= -1    pixel = 7,4
d= 7     pixel = 8,5
```