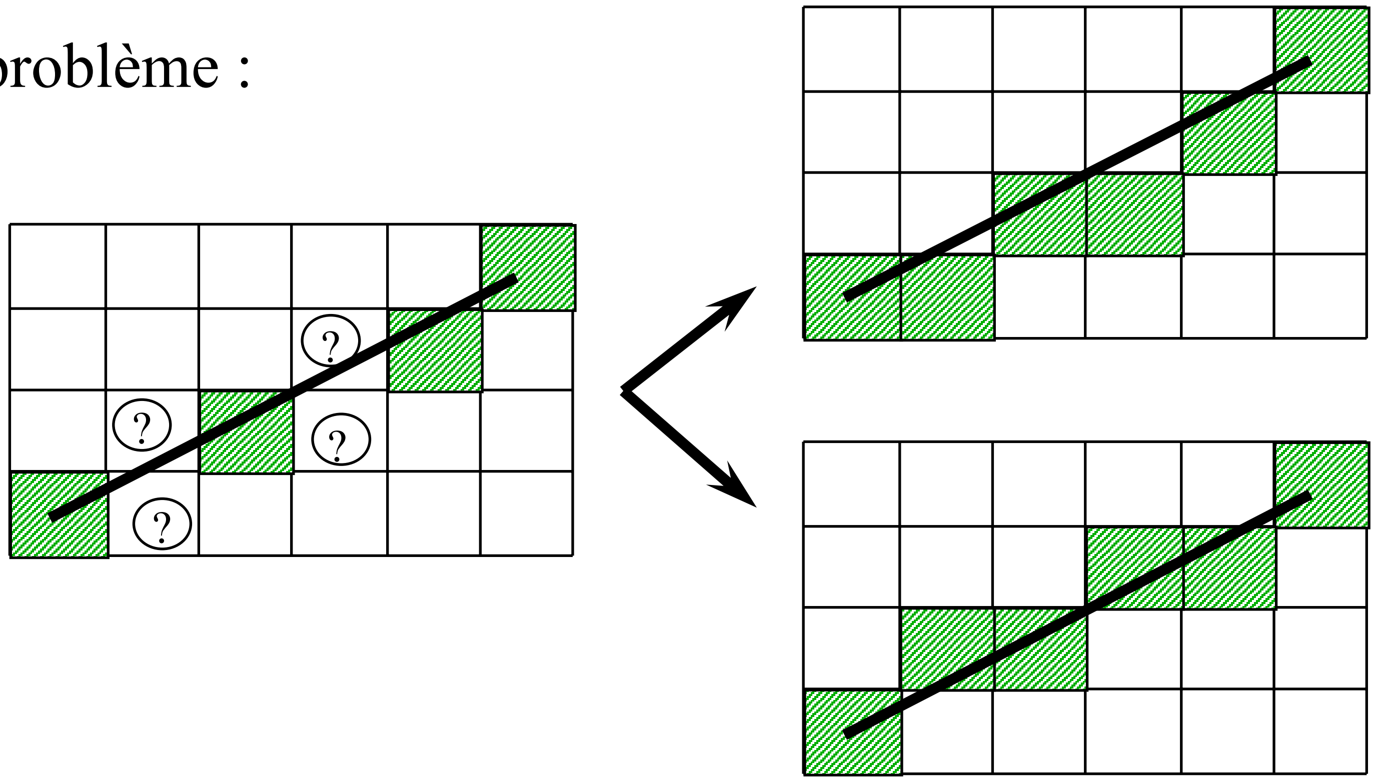
# ALGORITHMES 2D

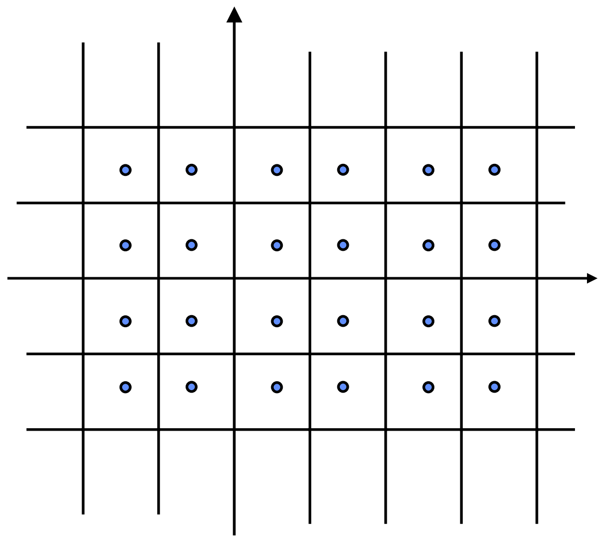
## 1 – Tracé de segments de droite

problème :

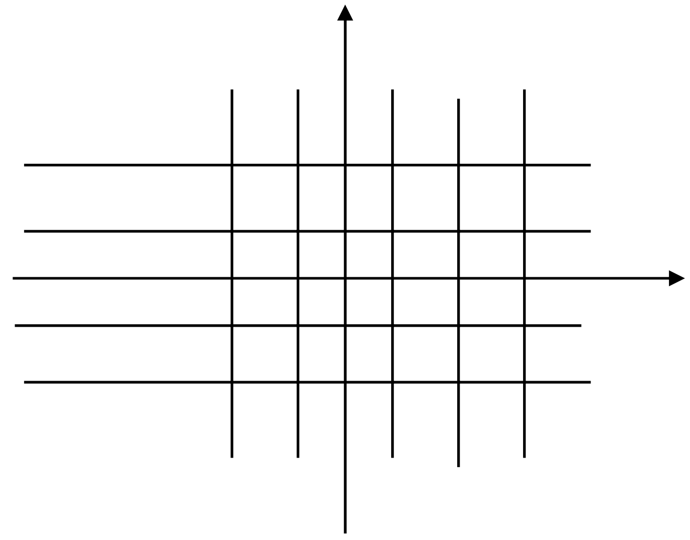


# 1.1. Solution naïve

```
fonction droite (x1, x2, y1, y2)

{  dx = x2 - x1 ;
   dy = y2 - y1 ;
   a = dy/dx ;
   b = y1 - a* x1 ;
   x  ←  x1 ;
   tant que  x < x2 faire
            {  x  ←    x+1 ;
               afficher(x, (int) (a*x+b)) ;
            }
}
```

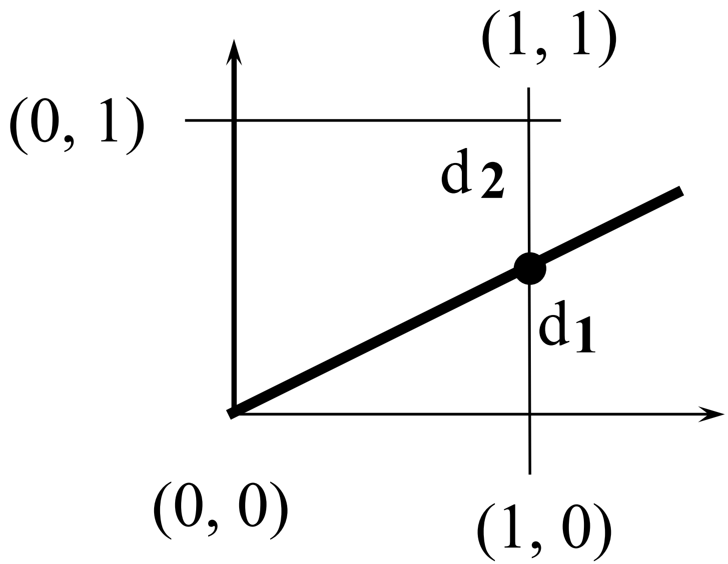


représentation des pixels



représentation de la grille des centres

# 1.2. Solution de Bresenham




principe :

si $d_1 < d_2$ alors afficher (1, 0)

sinon afficher (1, 1).

$e = d_1 - d_2$ est le **paramètre de décision**.

fonction BRESENHAM1 ($x_1$, $x_2$, $y_1$, $y_2$)

```
{ Δx ← x₂ - x₁ ; Δy ← y₂ - y₁ ;
   x ← x₁ ;  y ← y₁ ;
   e ← Δy / Δx - 1/2 ;
   afficher (x,y) ;
   tant que x < x₂ faire
          { x ← x+1 ;
          si e < 0 alors e ← e + 2 * Δy / Δx
          sinon
                 { e ← e + 2 * Δy / Δx - 2 ;
            y ← y + 1 ;
                 }
           afficher (x, y) ;
          }
}
```

## 1.1.3. Version entière

La version précédente utilise des calculs flottants.

Or ce qui est important est le signe de e.

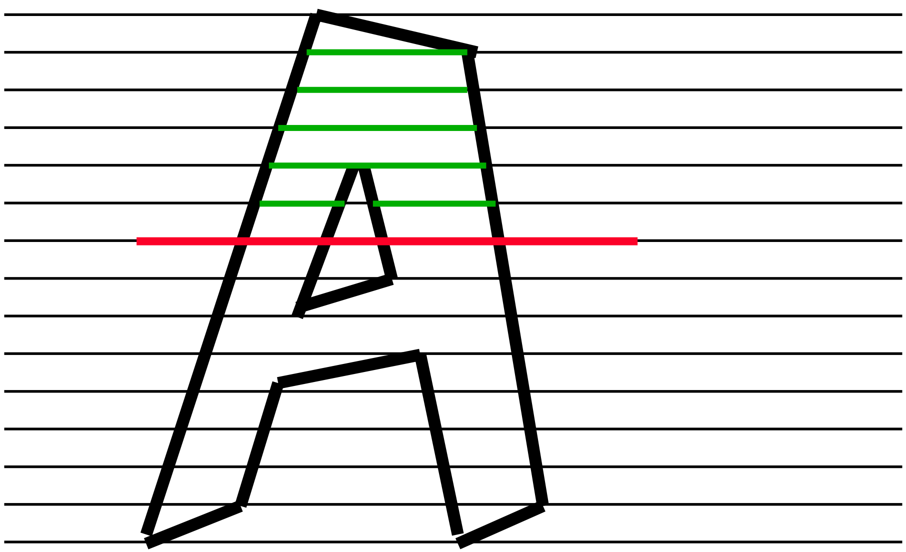
En posant $d = 2 * \Delta y - \Delta x$, on obtient la solution entière suivante :

```
fonction BRESENHAM2 (x1, x2, y1, y2)
{
   Δx ← x2 - x1 ;  Δy ← y2 - y1 ;
   x ← x1 ;   y ← y1 ;
   inc1 ← 2.Δy ;  inc2 ← 2(Δy - Δx) ;

   d ← (inc1+inc2)/2 ; afficher (x,y) ;

  tant que x < x2 faire
      { x ← x+1 ;
        si d < 0 alors d ← inc1 + d
        sinon
         { d ← d + inc2 ;
           y ← y + 1 ;
         }
        afficher (x,y) ;
      }
 }
```

# 2 – Remplissage de taches

On veut remplir une zone du plan dont la frontière est définie par un contour (polygonal ou non) dans le monde.

**principe de la méthode** : on différencie bords gauches et bords droits puis on déplace le long de ces bords les extrémités d'un segment horizontal.



```
programme principal ( )
{ pour chaque ligne i faire
  {
    déterminer l'ensemble E_i des arêtes qui coupent i ;
    pour chaque e ∈ E_i faire
          calculer e ∩ i ;
      trier les éléments de e ∩ i ;
      afficher les segments horizontaux ;
  }
```

si le contour est polygonal, on peut ne pas tout recommencer à chaque fois, en utilisant la cohérence des polygones et en réalisant des calculs **incrémentaux** :

$E_i$ = liste d'arêtes actives

$E_{i+1} = E_i$ - {arêtes qui se terminent sur ligne i}

+ {arêtes qui débutent sur ligne (i+1)}

si $x_i = e \cap i$ avec $e \in E_i$

$x_{i+1} = x_i + D_{xy}$ ,

où $D_{xy}$ = accroissement des abscisses de e entre les lignes i et i+1.

MAIS il existe des cas particuliers :