Synthèse d'Images Numériques

Espaces discrets

Philippe.Even@univ-lorraine.fr

Impact de la numérisation

```
double n = 0;
for (int i = 0; i < 100; i ++) n+= 1/100.;
System.out.println (n == 1.);
```

?

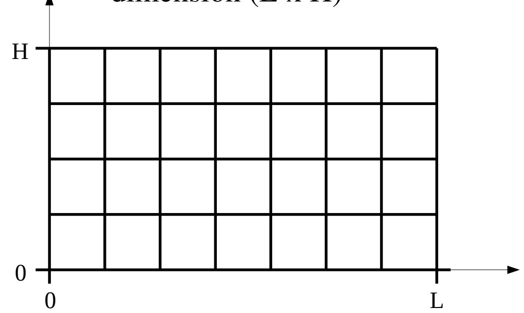
Espaces discrets

Deux approches duales :

> Partitionnement par une grille régulière
> Pavage par un réseau de cellules
- pixels dans le cas 2D
- voxels dans le cas 3D

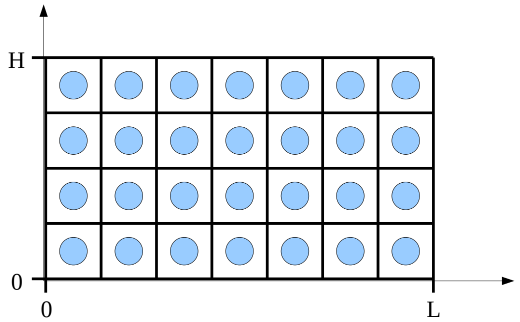
Espaces discrets

Partitionnement par une grille rectangulaire
dimension (L x H)



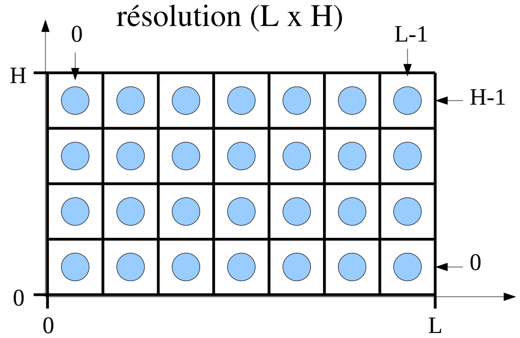
Espaces discrets

Pavage par un réseau de pixels



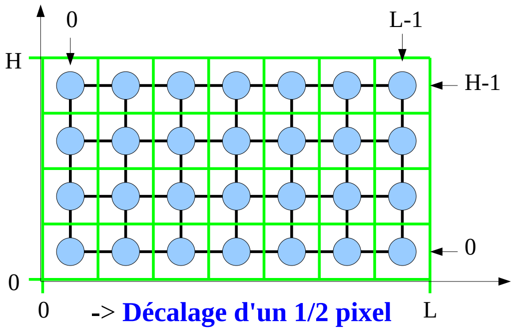
Espaces discrets

Pavage par un réseau de pixels
résolution (L x H)



Espaces discrets

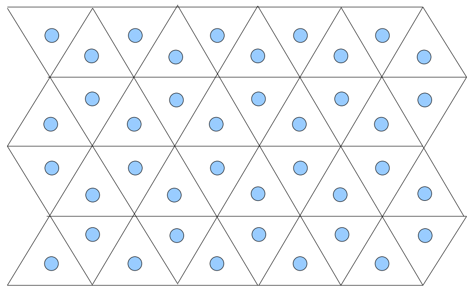
Grille rectangulaire : coordonnées grille / pixel



-> **Décalage d'un 1/2 pixel**

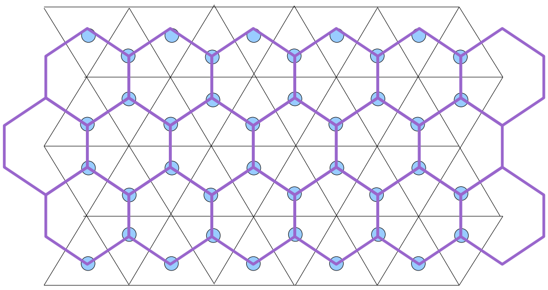
Espaces discrets

Grille triangulaire : **pavage** par des pixels



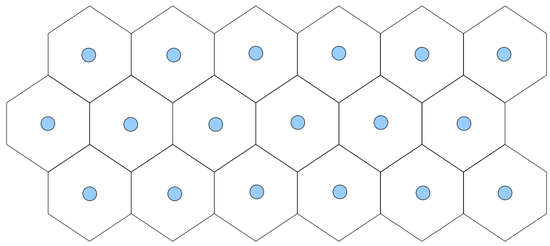
Espaces discrets

Grille triangulaire : **maillage** des centres



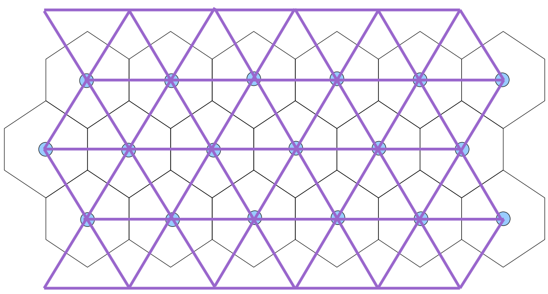
Espaces discrets

Grille hexagonale : **pavage** par des pixels



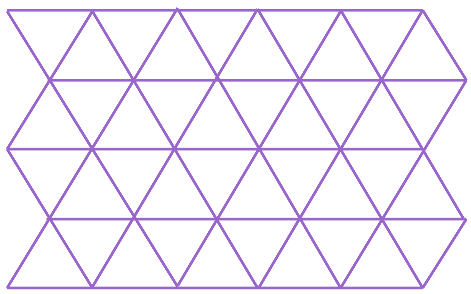
Espaces discrets

Grille hexagonale : **maillage** des centres



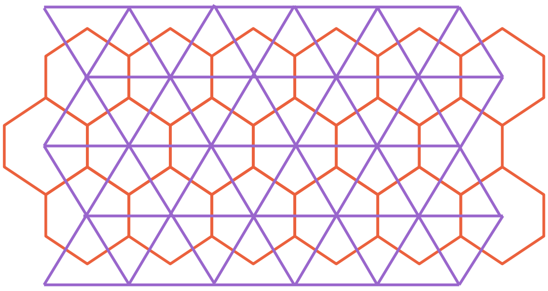
Espaces discrets

Grille hexagonale --> grille rectangulaire !



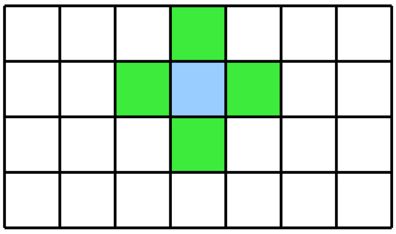
Espaces discrets

Grille hexagonale / triangulaire : dualité



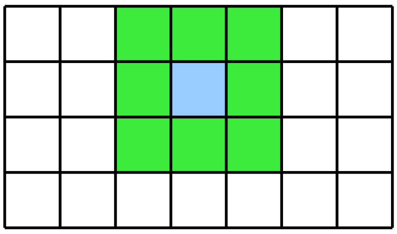
Aspects topologiques

Grille rectangulaire : 4 voisins par les arêtes
(ou 1-adjacence)



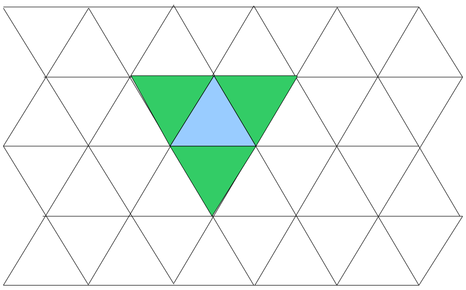
Aspects topologiques

Grille rectangulaire : 8 voisins par les sommets
(ou 0-adjacence)



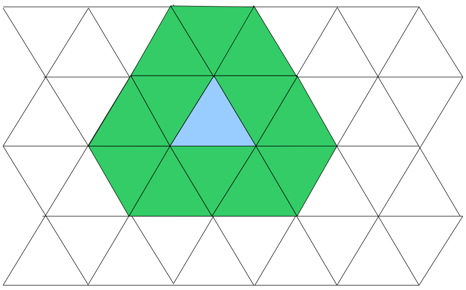
Aspects topologiques

Grille triangulaire : 3 voisins par les arêtes



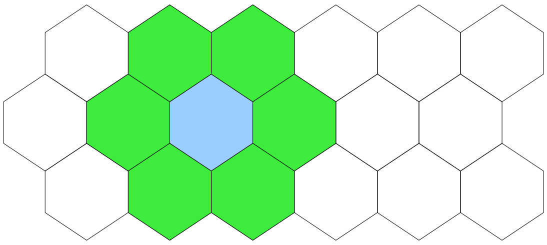
Aspects topologiques

Grille triangulaire : 12 voisins par les sommets



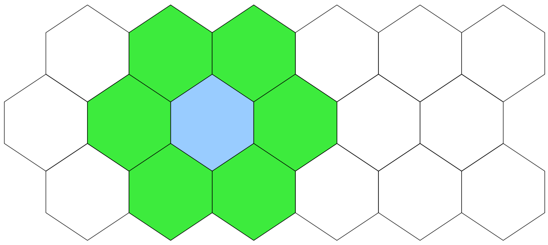
Aspects topologiques

Grille hexagonale : 6 voisins par les arêtes



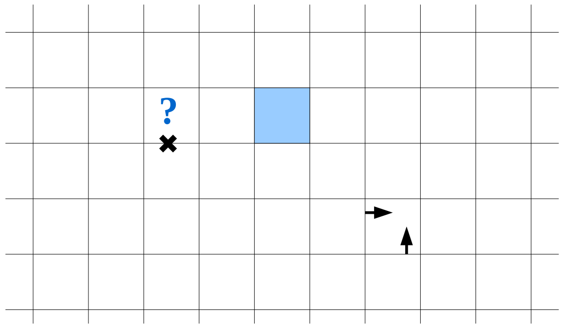
Aspects topologiques

Grille hexagonale : 6 voisins par les sommets

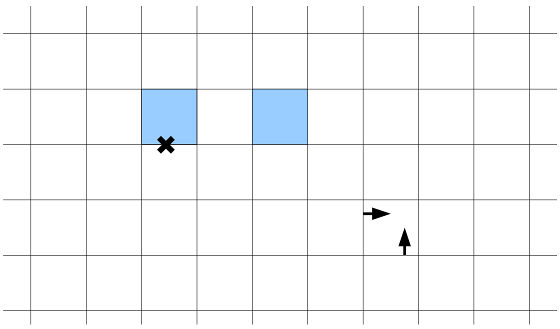


Discrétisation de points

Nécessité d'une convention pour les bords

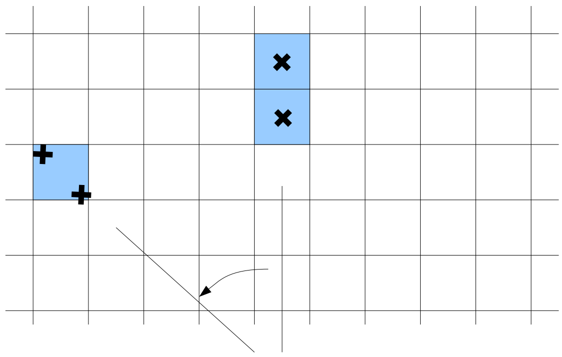


Discrétisation de points



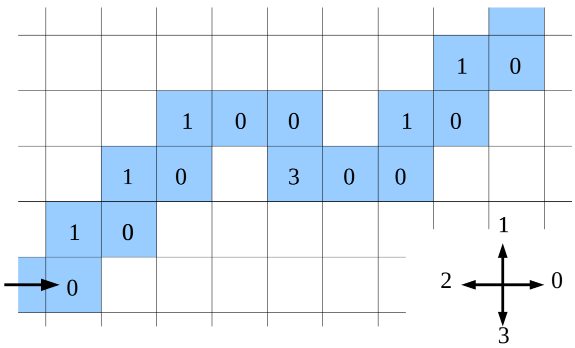
Discrétisation de points

Irréversibilité des rotations



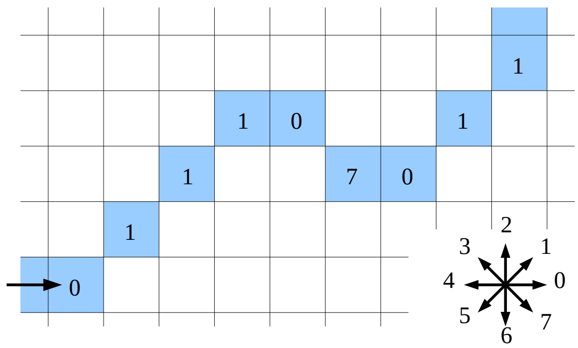
Discrétisation de courbes

Chemin : Code de Freeman en 1-adjacence



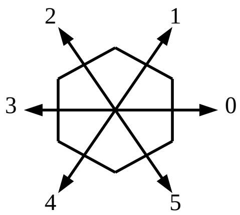
Discrétisation de courbes

Chemin : Code de Freeman en 0-adjacence



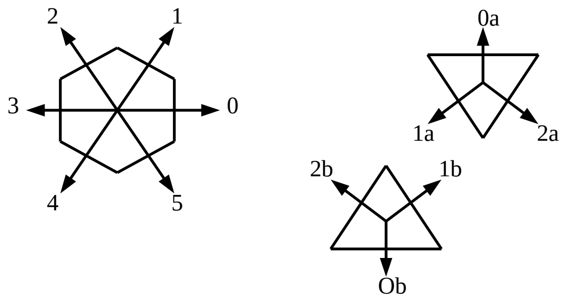
Discrétisation de courbes

Chemin : Code de Freeman en 1-adjacence pour les grilles hexagonale



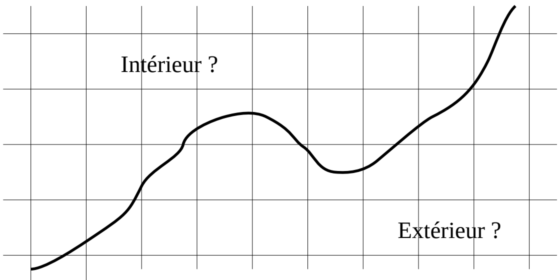
Discrétisation de courbes

Chemin : Code de Freeman en 1-adjacence pour les grilles hexagonale et triangulaire



Discrétisation de courbes

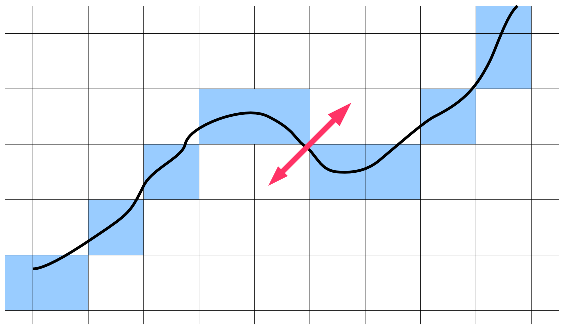
Courbe : chemin fermé



Critère de Jordan : séparation de l'espace

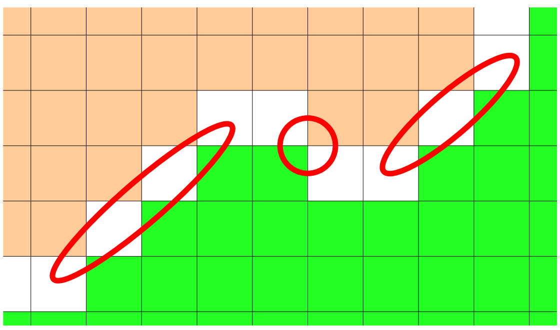
Discrétisation de courbes

Cas de la 0-adjacence : pas de séparation de l'espace !



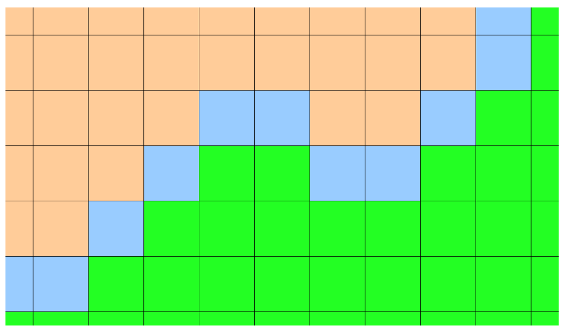
Discrétisation de courbes

En 1-adjacence : aucune courbe ne les sépare !



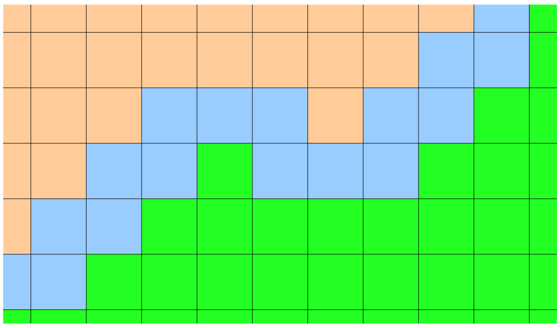
Discrétisation de courbes

Paires de Jordan (ex. : courbe en 0-adj - région en 1-adj)



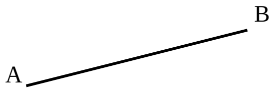
Discrétisation de courbes

Paires de Jordan (courbe en 1-adj - région en 0-adj)



Représentation de segments droits

Segment de droite AB, avec $A(X_A,Y_A)$ et $B(X_B,Y_B)$



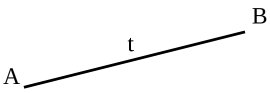
Droite support : $ax + by = c$

Pente $m = \frac{\Delta Y}{\Delta X} = - \frac{a}{b}$

$$\begin{cases} a = - \Delta Y = - (Y_B - Y_A) \\ b = \Delta X = X_B - X_A \\ c = aX_A + bY_A \end{cases}$$

Représentation de segments droits

Segment de droite AB, avec $A(X_A,Y_A)$ et $B(X_B,Y_B)$



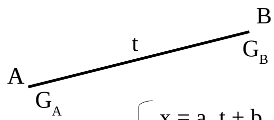
Equation paramétrique : $\begin{cases} x = a_1 t + b_1 \\ y = a_2 t + b_2 \end{cases}$

en A : $t = 0$ --> $b_1 = X_A$ et $b_2 = Y_A$

en B : $t = 1$ --> $a_1 = X_B - X_A$ et $a_2 = Y_B - Y_A$

Représentation de segments droits

Application à l'interpolation d'un paramètre



Equation paramétrique : $\begin{cases} x = a_1 t + b_1 \\ y = a_2 t + b_2 \end{cases}$

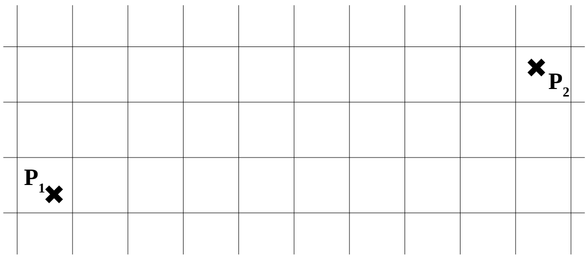
$$\boxed{\begin{array}{l} g = a_G t + b_G \\ \begin{cases} b_G = G_A \\ a_G = G_B - G_A \end{cases} \end{array}}$$

en A : $t = 0$ --> $b_1 = X_A$ et $b_2 = Y_A$

en B : $t = 1$ --> $a_1 = X_B - X_A$ et $a_2 = Y_B - Y_A$

Discrétisation de segments droits

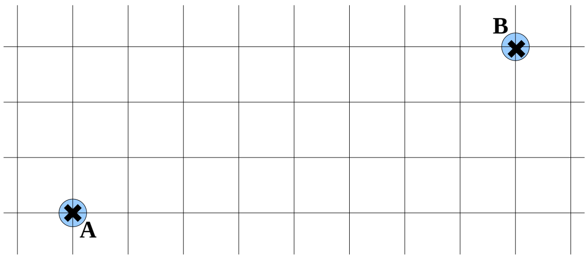
Discrétisation du segment de droite P_1P_2



Discrétisation de segments droits

Discrétisation du segment de droite P_1P_2

-> discrétisation du segment **disc(P_1) - disc(P_2) = AB**

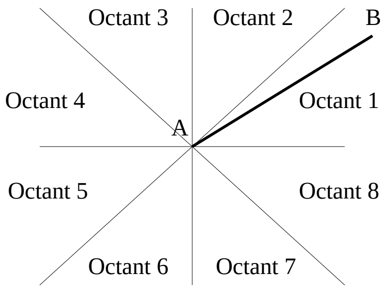


Discrétisation de segments droits

Raisonnement par octants

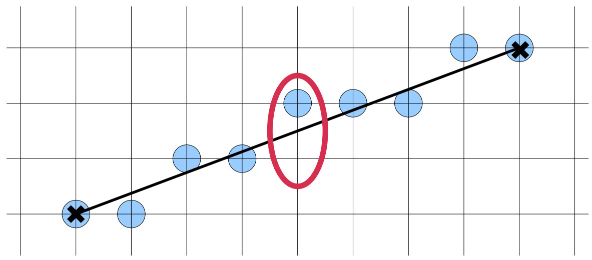
Selon la pente $\dfrac{\Delta Y}{\Delta X}$

$$\begin{cases} \Delta X = X_B - X_A \\ \Delta Y = Y_B - Y_A \end{cases}$$



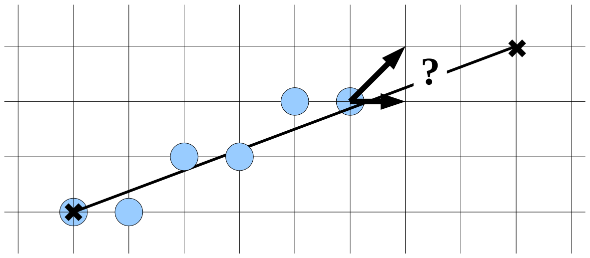
Discrétisation de segments droits

Algorithme des points médians :
- de X_A à X_B, numériser (x_i, y_i) --> $y_i = m\,x_i + c$



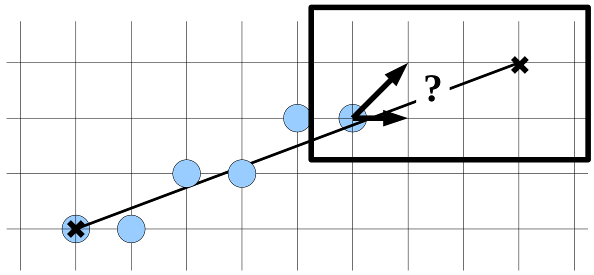
Discrétisation de segments droits

Algorithme de Bresenham : approche récursive
Où commence le palier suivant ?



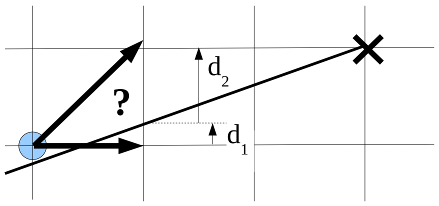
Discrétisation de segments droits

Algorithme de Bresenham
Où commence le palier suivant ?



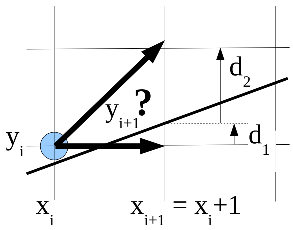
Discrétisation de segments droits

Algorithme de Bresenham
Critère de décision : $k = d_1 - d_2$



Discrétisation de segments droits

Algorithme de Bresenham
Critère de décision : $k = d_1 - d_2$
Equation de la droite : $y = mx+c$



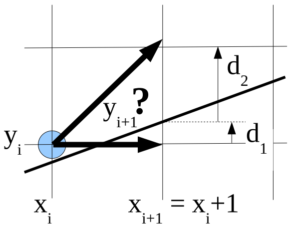
$$\begin{cases} d_1 = m(x_i + 1) + c - y_i \\ d_2 = y_i + 1 - m(x_i + 1) - c \end{cases}$$

Si $k < 0$: $y_{i+1} = y_i$
Sinon : $y_{i+1} = y_i + 1$

Discrétisation de segments droits

Algorithme de Bresenham

Critère de décision : $k = d_1 - d_2$



$$\begin{cases} d_1 = m(x_i + 1) + c - y_i \\ d_2 = y_i + 1 - m(x_i + 1) - c \end{cases}$$

$$k_i = 2\frac{\Delta Y}{\Delta X}(x_i+1) - 2y_i - 1 + 2c$$

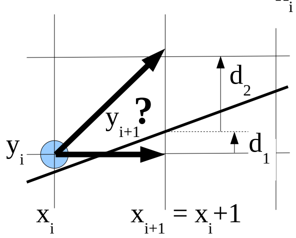
On remplace k par $k' = \Delta X\ k$

Discrétisation de segments droits

Algorithme de Bresenham

Critère de décision : $k' = \Delta x\ (d_1 - d_2)$

$$k_i' = 2\ \Delta Y\ (x_i+1) - \Delta X\ (2y_i + 1 - 2c)$$

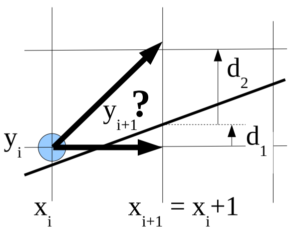


Peut-on déduire k_{i+1} à partir de k_i ?

Discrétisation de segments droits

Algorithme de Bresenham

$$k_i' = 2\ \Delta Y\ (x_i+1) - \Delta X\ (2y_i + 1 - 2c)$$



Si $k < 0$: $y_{i+1} = y_i$

et $\boxed{k_{i+1}' = k_i' + 2\ \Delta Y}$

Sinon : $y_{i+1} = y_i + 1$

et $\boxed{k_{i+1}' = k_i' + 2\ \Delta Y - 2\ \Delta X}$

On peut déduire k_{i+1} à partir de k_i !

Discrétisation de segments droits

Algorithme de Bresenham (démarrage)

$$k_i' = 2\ \Delta Y\ (x_i+1) - \Delta X\ (2y_i + 1 - 2c)$$

$$k_0' = 2\ \Delta Y\ (x_A+1) - \Delta X\ (2y_A + 1 - 2c)$$

or $c = y_A - mx_A \rightarrow \Delta X\ c = \Delta X\ y_A - \Delta Y\ x_A$

$$k_0' = 2\ \Delta Y\ (x_A+1) - \Delta X\ (2y_A + 1) + 2\ (\Delta X\ y_A - \Delta Y\ x_A)$$

$$\boxed{k_0' = 2\ \Delta Y - \Delta X}$$

Discrétisation de segments droits

Algorithme de Bresenham (algorithme pour l'octant 1)

```
i1  <-- 2 ΔY
i2  <-- 2 ΔY - 2 ΔX
k   <-- 2 ΔY - ΔX
x <-- X_A
y <-- Y_A
tracer (x, y)
Tant que (x < X_B)
        x ++
        Si k >= 0
                y ++
                k += i2
        Sinon
                k += i1
        tracer (x, y)
```

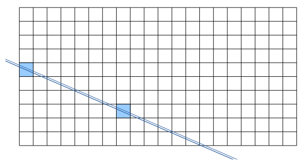

Discrétisation de segments droits

Algorithme arithmétique

-> basé sur la notion de droite discrète

$$0 \leq ax + by + c < \omega$$

épaisseur arithmétique (contrôle l'écart entre les droites support)



$0 \leq 3x + (-7)y + 35 < 1$

a, b, c, ω, x et y entiers

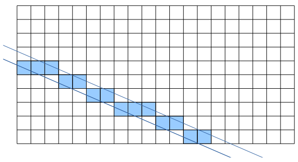
$$\begin{cases} a = \Delta Y\ /\ \mathrm{pgdc}\ (\Delta X, \Delta Y) \\ b = -\ \Delta X\ /\ \mathrm{pgdc}\ (\Delta X, \Delta Y) \end{cases}$$

Discrétisation de segments droits

Algorithme arithmétique

-> basé sur la notion de droite discrète

$0 \leq ax + by + c < \omega$



épaisseur arithmétique (contrôle l'écart entre les droites support)

a, b, c, ω, x et y entiers

$\begin{cases} a = \Delta Y \,/\, \mathrm{pgdc}\,(\Delta X, \Delta Y) \\ b = -\,\Delta X \,/\, \mathrm{pgdc}\,(\Delta X, \Delta Y) \end{cases}$

$0 \leq 3x + (-7)y + 35 < 7$

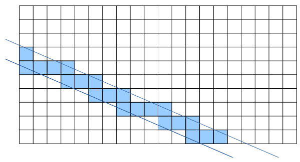
$\boxed{\omega = \max(|a|, |b|)}$: droite naïve (0-connexe)

Discrétisation de segments droits

Algorithme arithmétique

-> basé sur la notion de droite discrète

$0 \leq ax + by + c < \omega$



épaisseur arithmétique (contrôle l'écart entre les droites support)

a, b, c, ω, x et y entiers

$\begin{cases} a = \Delta Y \,/\, \mathrm{pgdc}\,(\Delta X, \Delta Y) \\ b = -\,\Delta X \,/\, \mathrm{pgdc}\,(\Delta X, \Delta Y) \end{cases}$

$0 \leq 3x + (-7)y + 35 < 10$

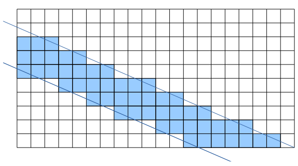
$\boxed{\omega = |a| + |b|}$: droite standard (1-connexe)

Discrétisation de segments droits

Algorithme arithmétique

-> basé sur la notion de droite discrète

$0 \leq ax + by + c < \omega$



épaisseur arithmétique (contrôle l'écart entre les droites support)

a, b, c, ω, x et y entiers

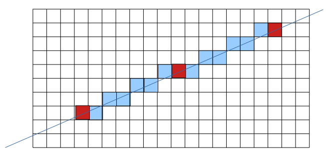
$\begin{cases} a = \Delta Y \,/\, \mathrm{pgdc}\,(\Delta X, \Delta Y) \\ b = -\,\Delta X \,/\, \mathrm{pgdc}\,(\Delta X, \Delta Y) \end{cases}$

$0 \leq 3x + (-7)y + 35 < 21$

Discrétisation de segments droits

Algorithme arithmétique

Principe : dans l'octant O1, quand on augmente de 1 en x, on augmente de $\Delta Y / \Delta X$ en y



Pour chaque position i :

$(i * \Delta Y) / \Delta X$ -> palier

$(i * \Delta Y) \% \Delta X$ -> hauteur

Discrétisation de segments droits

Algorithme de Bresenham

```
i1  <-- 2 ΔY
i2  <-- 2 ΔY - 2 ΔX
k   <-- 2 ΔY - ΔX
x <-- X_A
y <-- Y_A
tracer (x, y)
Tant que (x < X_B)
        x ++
        Si k >= 0
                y ++
                k += i2
        Sinon
                k += i1
        tracer (x, y)
```

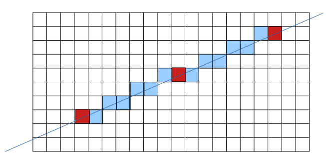
Algorithme arithmétique

```
r   <-- r_ini
x <-- X_A
y <-- Y_A
tracer (x, y)
Tant que (x < X_B)
        x ++
        r += ΔY
        Si r >= ΔX
                y ++
                r -= ΔX
        Sinon
                r += ΔY
        tracer (x, y)
```


Discrétisation de segments droits

Algorithme arithmétique

Avantage sur Bresenham, un meilleur contrôle de la hauteur du segment



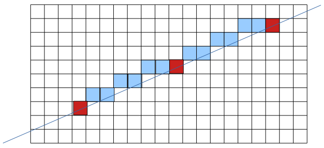
Discrétisation basée-frontière :

r_ini = $\Delta X / 2$

Discrétisation de segments droits

Algorithme arithmétique

Avantage sur Bresenham : meilleur contrôle de la hauteur du segment

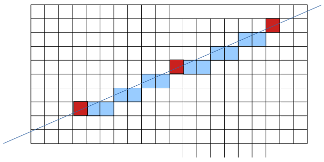


Discrétisation basée-objet sup :
r_ini = 0

Discrétisation de segments droits

Algorithme arithmétique

Avantage sur Bresenham : meilleur contrôle de la hauteur du segment



Discrétisation basée-objet inf :
r_ini = ΔX - 1