TRACÉ DE DROITES, FRACTIONS CONTINUES ET MORPHISMES ITÉRÉS

PAR

JEAN BERSTEL

Résumé. — Tracer une droite sur une grille de pixels revient à allumer les pixels les plus proches de la droite réelle. Si l'on repère les déplacements entre deux pixels consécutifs, on obtient un mot sur deux lettres, disons a et b, qui est entièrement déterminé par la pente de la droite.

Le calcul de ce mot peut se faire de plusieurs manières, faisant intervenir les fractions continues et l'itération de certains morphismes. Les mots obtenus sont sturmiens.

1. Tracer un segment de droite

Le tracé d'un segment de droite sur un écran graphique se fait en allumant les pixels de l'écran les plus proches de la droite réelle, la notion de proximité pouvant d'ailleurs varier.

On peut, sans perte de généralité, supposer que l'on trace le segment de droite de l'origine à un point à coordonnées entières (u, v), et, par un argument de symétrie, on peut supposer $0 < v < u$. On peut également supposer que u et v sont premiers entre eux, car sinon le segment s'obtient en répétant le tracé de l'origine au point $(u/\operatorname{pgcd}(u, v), v/\operatorname{pgcd}(u, v))$.

Avec ces conventions, il s'agit d'allumer un pixel d'abscisse x pour chaque entier $x \in \{0, \dots, u\}$, et de déterminer l'ordonnée $y(x)$ correspondante. Il résulte de nos hypothèses que pour deux abscisses consécutives x, $x+1$, on a

$$y(x+1) = \begin{cases} y(x) & \text{si } \left[(x+1)\dfrac{v}{u}\right] = \left[x\dfrac{v}{u}\right], \\ y(x)+1 & \text{si } \left[(x+1)\dfrac{v}{u}\right] = 1 + \left[x\dfrac{v}{u}\right]. \end{cases}$$

Nous codons ces deux situations respectivement par les lettres a et b, et obtenons donc un *mot* $c_1 c_2 \dots c_n \in \{a,b\}^*$ avec

$$c_x = \begin{cases} a & \text{si } y(x) = y(x-1), \\ b & \text{si } y(x) = 1 + y(x-1), \end{cases} \qquad (1 \le x \le n).$$

Il existe de nombreuses stratégies pour choisir et calculer la fonction $y(x)$ (voir Foley, Van Dam [2], Newman, Sproull [4], Hégron [3]). Nous en résumons trois qui nous intéresserons par la suite.

a) *Méthode du meilleur ajustement.* On choisit, pour chaque x, l'entier $y(x)$ le plus proche du point d'abscisse x sur la droite. La Figure 1 montre le résultat que l'on obtient dans ce cas.

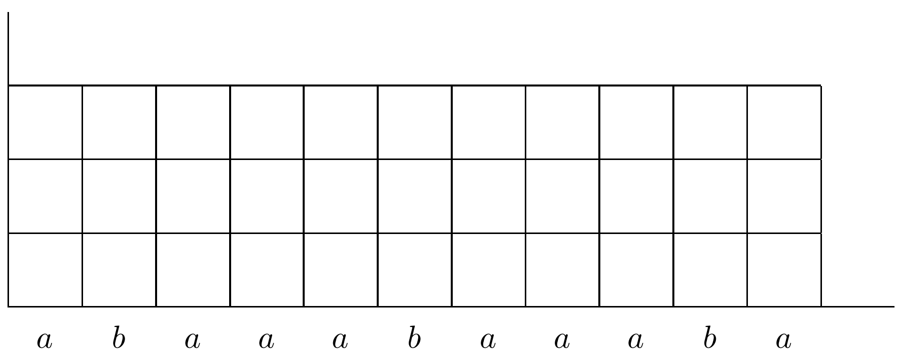

Fig. 1

Pour obtenir le mot associé, on adapte l'algorithme classique de Bresenham (voir [2]) qui n'utilise que des calculs sur des entiers, comme suit :

```
d := 2 * v - u;
for x := 1 to u do
    if d > 0
    then begin d := d + 2 * v - 2 * u ; write(b) end
    else begin d := d + 2 * v ; write(a) end;
```

b) *Méthode* “dda”. On suit la droite réelle en effectuant des déplacements horizontaux tant que le pixel allumé est strictement au-dessus de la droite et des déplacements diagonaux si le pixel est au-dessous. Voir Figure 2.

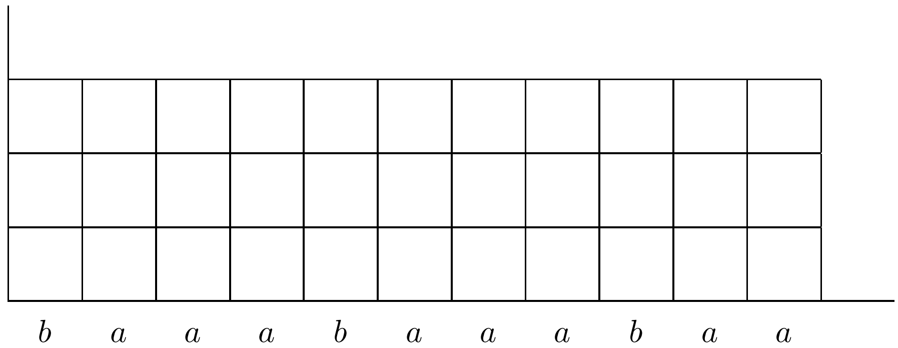

Fig. 2

Le mot associé s’obtient, lui aussi, par un algorithme court :

$d := 0\,;$
for $x := 1$ **to** u **do**
    if $d \geq 0$
    then begin $d := d + v - u\,;\ \mathit{write}(b)$ **end**
    else begin $d := d + v\,;\ \mathit{write}(a)$ **end**

Les mots obtenus par ces deux méthodes ne sont pas les mêmes, mais ils sont conjugués, comme nous allons le voir.

c) *Méthode des parties entières.* On prend pour $y(x)$ le point à coordonnées entières se situant au-dessous de la droite. Voir Figure 3. Du point de vue graphique, cette méthode donne de bons résultats si le point calculé est le coin inférieur gauche du pixel qui est supposé remplir le carré.

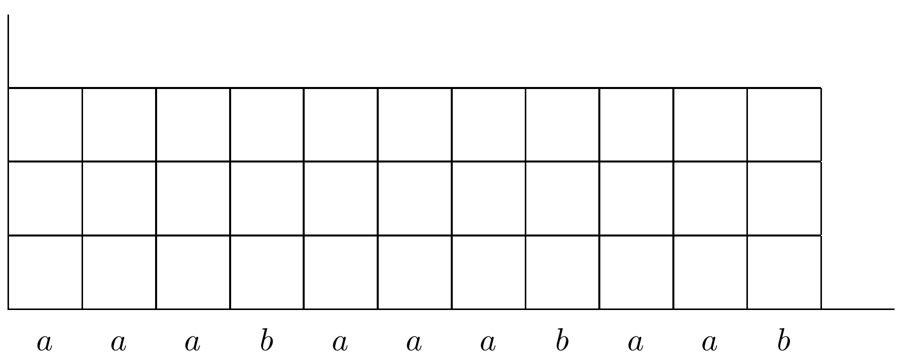

Fig. 3

Le programme correspondant s'écrit immédiatement :

$d := 0\,;$
for $x := 1$ **to** u **do**
  if $d \le u$
  then begin $d := d + v\,;\ write(a)$ **end**
  else begin $d := d - u\,;\ write(b)$ **end**

On appellera *mot de Christoffel* le mot obtenu de cette manière, car d'après Venkov [8], il a été l'un des premiers à étudier ces mots.

Voici une première observation :

Observation. — *Soient w, w' et w'' les mots obtenus par la méthode du meilleur ajustement, la méthode* dda *et la méthode des parties entières pour un même point (u, v). Ces trois mots sont conjugués, et plus précisément*

$$w'b = bw'', \quad sw' = ws,$$

où s est le facteur droit de w'' dont la longueur ℓ est solution de la congruence $v\ell + [u/2] \equiv 0 \mod u$.

2. Calculs directs

Il existe des méthodes de calcul des divers mots associés à une droite qui ne sont pas incrémentales comme les algorithmes de la section précédente. Le premier détermine le mot du meilleur ajustement, le deuxième les autres.

a) *Algorithme de* Pitteway, Earnshaw [5]. Cet algorithme étrange calcule le mot du meilleur ajustement, en calculant deux colonnes de mots. Chaque mot d'une colonne est obtenu par une opération combinatoire à partir des mots de la ligne précédente. Voici l'algorithme :

$u := u - v\,;\ s := a\,;\ t := b\,;$
while $u \ne v$ **do**
  if $u > v$
  then begin $u := u - v\,;\ t := s \cdot \tilde{t}$ **end**
  else begin $v := v - u\,;\ s := t \cdot \tilde{s}$ **end**;
  résultat $:= t \cdot \tilde{s}$

Les mots mis à jour sont s et t ; le point désigne la concaténation et $\tilde{t}$ est le mot miroir de t. On peut montrer qu'à chaque étape du calcul, les mots s et t sont des *palindromes faibles*, *i.e.*, des mots de la forme $y\ z\ \tilde{y}$ où z est un mot de longueur au plus 2. Il en est donc de même pour le mot résultat. Voici un exemple du fonctionnement de l'algorithme.

Exemple. — $u = 11$, $v = 3$

v	u	s	t
3	11		
3	8	a	b
3	5	a	ab
3	2	a	aba
1	2	$abaa$	aba
1	1	$abaa$	$abaaaba$
résultat		$abaaabaaaba$	

b) *Algorithme pour les mots de Christoffel.* Dans ce cas, l'algorithme est légèrement plus simple. Il s'énonce comme suit :

$u := u - v\,;\ s := a\,;\ t := b\,;$
while $u \neq v$ **do**
  if $u > v$
  then begin $u := u - v\,;\ t := s \cdot t$ **end**
  else begin $v := v - u\,;\ s := s \cdot t$ **end**;
  résultat $:= s \cdot t$

Les mots s et t sont donc, dans ce cas, simplement multipliés. Rappelons que pour la méthode dda, on obtient le mot correspondant en remplaçant la dernière ligne par

$$\text{résultat} := b \cdot st \cdot b^{-1}.$$

Exemple. — $u = 11$, $v = 3$: voir page suivante.

Nous donnerons une version plus compacte de cet algorithme plus loin.

3. Mots sturmiens

Nous avons associé au tracé d'un segment de droite un mot. Ces mots de droites ne sont bien sûr pas quelconques, et ils ont en fait été étudiés en détail. Nous rappelons ici quelques unes de leurs propriétés. Pour un exposé plus complet, on peut consulter Coven, Hedlund [1] ou la présentation de Rauzy [6].

v	u		
3	11	s	t
3	8	a	b
3	5	a	ab
3	2	a	aab
1	2	$aaab$	aab
1	1	$aaab$	$aaabaab$
résultat		$aaabaaabaab$	

Algorithme pour les mots de Christoffel : $u = 11$, $v = 3$

La façon la plus visuelle de définir un mot de Sturm est de tracer une droite bi-infinie dans le plan, et de repérer la suite de ses intersections avec les droites horizontales et verticales de la grille entière. Une intersection horizontale est notée par un a, une intersection verticale par un b, une intersection dans un coin par ab. On obtient ainsi un mot bi-infini qui est périodique si et seulement la pente de la droite est un nombre rationnel. Voir Figure 4.

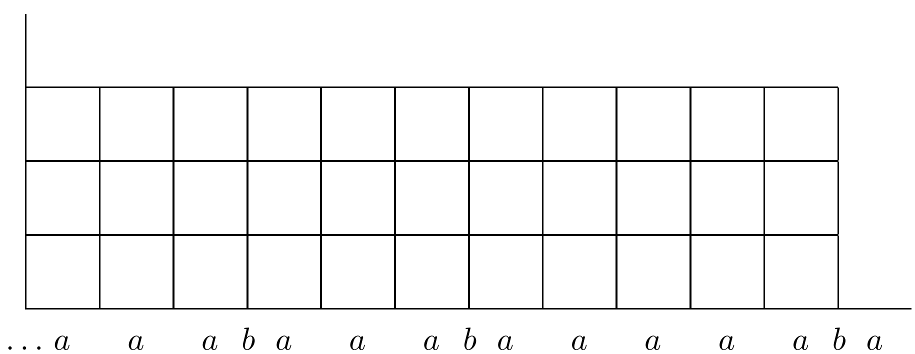

Fig. 4

Si l'on remplace chaque facteur ab par un b, on obtient le mot qui nous intéresse.

Rappelons deux caractérisations fort intéressantes des mots sturmiens :

Caractérisation 1. — *Un mot est sturmien si et seulement si pour tout entier n inférieur à sa période, il a exactement $n+1$ facteurs de longueur n.*

Exemple. — Le mot *aaabaaabaab* a exactement 7 facteurs de longueur 6, à condition de les prendre circulairement. Ce sont :

$$aaabaa,\quad aabaaa,\quad abaaab,\quad baaaba,\quad aabaab,\quad abaaba,\quad baabaaa.$$

CARACTÉRISATION 2. — *Un mot est sturmien si et seulement si, pour tout n, et pour tous facteurs w, w' de longueur n, on a*

$$\big|\, |w|_a - |w'|_a \big| \le 1.$$

Exemple. — Dans l'exemple ci-dessus, les 7 mots de longueur 6 contiennent en effet 4 ou 5 fois la lettre a.

COROLLAIRE. — *Si, dans un mot sturmien, on remplace les facteurs ab par b, en laissant le reste inchangé, on obtient encore un mot sturmien.*

Comme conséquence, nous obtenons donc que les mots de Christoffel sont sturmiens.

4. Constructions

Dans cette section, nous décrivons deux constructions explicites des mots de Christoffel. Partant de deux entiers u, v, premiers entre eux, avec $0 \le v < u$, nous cherchons donc le mot $E(v/u) = d_0 d_1 \cdots d_{u-1}$ avec

$$d_i = \begin{cases} a & \text{si } \left[(i+1)\dfrac{v}{u}\right] = \left[i\dfrac{v}{u}\right], \\ b & \text{si } \left[(i+1)\dfrac{v}{u}\right] = 1 + \left[i\dfrac{v}{u}\right]. \end{cases}$$

Par exemple, $E(3/11) = aaabaaabaab$.

a) *Fractions continues.* Si z est un nombre rationnel non négatif, et si

$$z = u_0 + \cfrac{1}{u_1 + \cfrac{1}{u_2 + \ddots + \cfrac{1}{u_n}}}$$

nous écrivons

$$z = [u_0\,;\ u_1, u_2, \ldots, u_n].$$

Comme nous nous intéressons à des nombres plus petits que 1, on aura $u_0 = 0$. Par exemple,

$$\frac{3}{11} = [0\,;\ 3, 1, 2].$$

b) *Algorithme descendant.* Cet algorithme n'est qu'une version compactifiée de la méthode des parties entières exposée en section 1.c). Pour

$$z = [0\,;\; u_1, u_2, \ldots, u_n],$$

on note

$$z_i = [0\,;\; u_1, u_2, \ldots, u_i] \qquad (0 \leq i \leq n)$$

la réduite d'ordre i. On a alors

$$E(z_0) = a, \quad E(z_1) = a^{u_1 - 1}b,$$

et les formules de récurrence

$$E(z_i) = \begin{cases} E(z_{i-1})^{u_i}\, E(z_{i-2}) & \text{si } i \text{ est impair,} \\ E(z_{i-2})\, E(z_{i-1})^{u_i} & \text{si } i \text{ est pair.} \end{cases}$$

Voici quelques exemples :

(i) pour $3/11 = [0\,;\; 3, 1, 2]$, on obtient successivement

$$\begin{aligned} E(0) &= a, \\ E(1/3) &= aab, \\ E(1/4) &= a \cdot aab, \\ E(3/11) &= aaab \cdot aaab \cdot aab. \end{aligned}$$

(ii) pour $5/12 = [0\,;\; 2, 2, 2]$, on calcule

$$\begin{aligned} E(0) &= a, \\ E(1/2) &= ab, \\ E(2/5) &= a \cdot ab \cdot ab, \\ E(5/12) &= aabab \cdot aabab \cdot ab, \end{aligned}$$

Rappelons, à propos de cet algorithme, les formules de calcul des réduites. En posant

$$z_i = \frac{p_i}{q_i} \qquad i = 0, \ldots, n$$

où p_i, q_i sont premiers entre eux, on a

$$\begin{aligned} p_0 &= 0, \quad p_1 = 1, \quad\ p_i = p_{i-1}u_i + p_{i-2}\,; \\ q_0 &= 1, \quad q_1 = u_1, \quad q_i = q_{i-1}u_i + q_{i-2}. \end{aligned}$$

On obtient, en retour, ces formules à partir de l'algorithme descendant en observant que, par définition des mots de Christoffel,

$$\begin{aligned} q_i &= \text{ longueur du mot } E(z_i), \\ p_i &= \text{“hauteur” du mot } E(z_i) = \text{nombre de } b \text{ dans } E(z_i). \end{aligned}$$

c) *Méthode ascendante.* Cette méthode est basée sur l'itération de deux morphismes qui sont

$$\alpha : \begin{cases} a \longrightarrow a \\ b \longrightarrow ab \end{cases} \qquad \beta : \begin{cases} a \longrightarrow ab \\ b \longrightarrow b \end{cases}$$

De façon mnémotechnique, α augmente le nombre de a et β augmente le nombre de b dans un mot. Soit $z = [0\,;\ u_1, u_2, \ldots, u_n]$. On a alors :

$$E(z) = \alpha^{u_1 - 1} \circ \beta^{u_2} \circ \alpha^{u_3} \circ \cdots \circ \gamma^{u_n - 1}(ab),$$

où $\gamma = \alpha$ ou $\gamma = \beta$ selon l'imparité de n.

Reprenons les exemples précédents :

(i) pour $3/11 = [0\,;\ 3, 1, 2]$, on obtient :

$$\begin{aligned} E\left(\frac{3}{11}\right) &= \alpha^2 \beta \alpha(ab) = \alpha^2 \beta(aab) = \alpha^2(ababb) \\ &= \alpha(aabaabab) = aaabaaabaab. \end{aligned}$$

(ii) pour $5/12 = [0\,;\ 2, 2, 2]$, le calcul donne

$$\begin{aligned} E\left(\frac{5}{12}\right) &= \alpha\beta^2\alpha(ab) = \alpha\beta^2(aab) = \alpha\beta(ababb) = \alpha(abbabbb) \\ &= aababaababab. \end{aligned}$$

d) *Comparaison.* Il y a un lien étroit entre les deux algorithmes. Pour l'exposer, il est plus commode de revenir à la version additive de l'algorithme descendant, telle qu'énoncée en section 1.c). Les deux méthodes, appliquées au nombre 9/13, donnent alors le tableau suivant

v	u			
9	13	s	t	x
9	4	a	b	$abbabbabbabbb$
$\vert$		$\downarrow$		$\uparrow \beta$
5	4	ab	b	$abababbb$
$\vert$		$\downarrow$		$\uparrow \beta$
1	4	abb	b	$aaab$
	$\vert$		$\downarrow$	$\uparrow \alpha$
1	3	abb	$abbb$	$aaab$
	$\vert$		$\downarrow$	$\uparrow \alpha$
1	2	abb	$abbabbb$	aab
	$\vert$		$\downarrow$	$\uparrow \alpha$
1	1	abb	$abbabbabbb$	ab
résultat		$abbabbabbabbb$		

Théorème. — *En substituant, à chaque ligne, dans le mot x, le mot s aux lettres a, et le mot t aux lettres b, on obtient le mot résultat.*

Par exemple, pour $x = aaaab$, on a $s = abb$ et $t = b$, et le mot résultat est bien égal à s^4t.

L'invariant ainsi décrit n'est pas sans rappeler la construction matricielle de développement en fraction continue, telle qu'étudiée en profondeur par Raney [6].

Rappelons que l'on considère deux matrices

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix},$$

et qu'à chaque développement en fraction continue $z = [0\,;\ u_1, u_2, \ldots, u_n]$ on associe la matrice

$$M(z) = A^{u_1-1}B^{u_2}A^{u_3}\ldots C^{u_n-1}$$

avec $C = A$ ou $C = B$ selon l'imparité de n. On a alors, pour $z = v/u$,

$$\begin{pmatrix} v \\ u-v \end{pmatrix} = M(z)\begin{pmatrix} 1 \\ 1 \end{pmatrix},$$

et, en particulier,

$$(1\ 1)\, M(z) \begin{pmatrix} 1 \\ 1 \end{pmatrix} = u.$$

Considérons maintenant une factorisation quelconque

$$M(z) = GD$$

où G et D sont formées de produits de matrices A et B. Alors

$$(1\ 1)\, G = (p\ q), \qquad \begin{pmatrix} k \\ \ell \end{pmatrix} = D \begin{pmatrix} 1 \\ 1 \end{pmatrix},$$

et, d'après ce que nous venons de voir, $(p, q)\binom{k}{\ell} = u$. L'invariant donné plus haut permet de donner l'interprétation suivante des nombres p, q, k et ℓ. En considérant la ligne correspondante du tableau comparatif, on a :

$$\begin{aligned} p &= \text{longueur du mot } s, & k &= \text{nombre de } b \text{ dans le mot } x, \\ q &= \text{longueur du mot } t, & \ell &= \text{nombre de } a \text{ dans le mot } x. \end{aligned}$$

Dans l'exemple précédent du nombre $z = 9/13$, on a

$$M(z) = BBAAA$$

et on obtient les suites de vecteurs suivantes

$$(1,1) \xrightarrow{B} (2,1) \xrightarrow{B} (3,1) \xrightarrow{A} (3,4) \xrightarrow{A} (3,7) \xrightarrow{A} (3,10)$$

$$\begin{pmatrix} 4 \\ 9 \end{pmatrix} \xleftarrow{B} \begin{pmatrix} 4 \\ 5 \end{pmatrix} \xleftarrow{B} \begin{pmatrix} 4 \\ 1 \end{pmatrix} \xleftarrow{A} \begin{pmatrix} 3 \\ 1 \end{pmatrix} \xleftarrow{A} \begin{pmatrix} 2 \\ 1 \end{pmatrix} \xleftarrow{A} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Remerciements : Je remercie Danièle Beauquier, François Bergeron, Srecko Brlek, Christine Duboc et Christophe Reutenauer pour de fructueuses discussions pendant la préparation de ce travail.

BIBLIOGRAPHIE

[1] Coven (E. M.), Hedlund (G. A.). — Sequences with Minimal Block Growth, *Math. Systems Theory*, t. **7**, 1973, p. 138–153.

[2] Foley (J. D.), Van Dam (A.). — *Fundamentals of Interactive Computer Graphics*. — Addison–Wesley, 1982.

[3] Hegron (G.). — *Synthèse d'image : algorithmes élémentaires*. — Dunod, 1985.

[4] Newman (W. M.), Sproull (R. F.). — *Principles of Interactive Computer Graphics*. — McGraw–Hill, 1979.

[5] Pitteway (M.L.V.), Earnshaw (R.A.). — Euclid's Algorithm and Line Drawing, [R. A. Earnshaw (ed)], *Fundamental Algorithms in Computer Graphics*, p. 101–106. — Springer–Verlag, 1985.

[6] Raney (G.N). — On Continued Fractions and Finite Automata, *Math. Ann.*, t. **206**, 1973, p. 265–283.

[7] Rauzy (G.). — Mots infinis en arithmétique, [M. Nivat, D. Perrin (eds)], *Automata on Infinite Words*, [Lecture Notes in Computer Science], pp. 165–171. — Springer–Verlag, vol. 192, 1985.

[8] Venkov (B.A.). — *Elementary Number Theory*. — Wolters–Noordhoff, 1970.