TP Récursivité : Fractales

1 Introduction

Les fractales sont définies par un procédé récursif de fabrication où un segment est remplacé par un motif composé lui-même de plusieurs segments de tailles inférieures. Chacun de ces nouveaux segments est á nouveau remplacé par le meme motif et ainsi de suite. Chaque itération supplémentaire correspond á une nouvelle courbe, de rang supérieur.

2 Mise en place du Module Graphique

Avant de commencer il faut mettre en place l'environnement graphique :

```
#load "graphics.cma";;
open Graphics;;
open_graph "";;
```

En OCaml la bibliothèque graphique s'inspire du principe de l'écran magique ; on contrôle un point courant á partir duquel on trace des segments en le deplaçant.

Voici les fonctions que vous allez devoir utiliser pour afficher vos fractales :

`val clear_graph : unit -> unit`

Efface le contenu de la fenetre.

`val moveto : x:int -> y:int -> unit`

Positionne le point courant sans tracer de segment.

`val rmoveto : dx:int -> dy:int -> unit`

Positionne le point courant en le déplacant de dx dy sans tracer de segment.

```
val lineto : x:int -> y:int -> unit
```
Trace un segment du point courant au point x y et positionne le point courant en x y.

```
val rlineto : dx:int -> dy:int -> unit
```
Trace un segment du point courant au point courant deplace de dx dy
et le positionne a ces nouvelles coordonnees

```
val set_color : c:color -> unit
```
Change la couleur courante

2.1 Exemple

```
clear_graph ();;
set_color red;;
moveto5050;;
lineto250250;;
```

Pour l'ensemble des fonctions recursives qui seront á implémenter on utilisera du pattern matching.

3 Montagnes

On désire générer aléatoirement une "montagne" selon le principe suivant :

- **étape 0** On trace un segment entre 2 points quelconques.
- **étape 1** On calcule une nouvelle hauteur aléatoire pour le milieu du segment (on pourra utiliser la fonction Random.int -> int -> int qui renvoie un int generé aleatoirement et borné par celui donné en argument).
- **étape n** On applique le meme processus sur chacun des nouveaux segments de l'étape précédente.

Processus : La "courbe" d'ordre n entre les points (x, y) et (u, v) est :

- $n = 0$ le segment [(x, y), (u, v)]
- $n \neq 0$ la courbe d'ordre $n - 1$ entre (x, y) et (m, h) suivie de la courbe d'ordre $n - 1$ entre (m, h) et (u, v), oú m est le "milieu" de x et u et h une hauteur calculée aléatoirement.

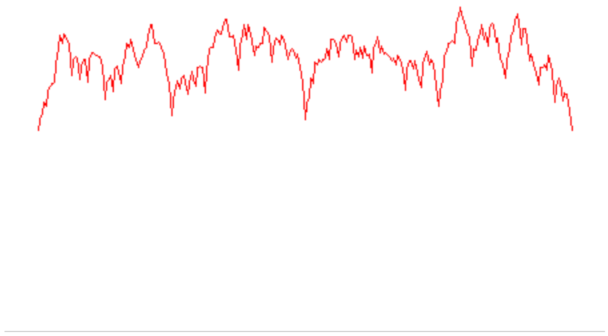


FIGURE 1 – Montagne rang 5

La fonction `mount` a donc le profil suivant : `mount -> int*int -> int*int -> int -> unit` prennant donc en paramètre les deux points (x, y) et (u,v) ainsi que l'ordre n.

Conseil : on peut calculer la nouvelle hauteur en fonction des 2 points et éventuellement de n. On peut, par exemple, diminuer la différence de hauteur au fur et á mesure que les points se rapprochent avec la formule suivante :

$$h = (y + v)/2 + Random.int(10n) \tag{1}$$

4 Dragon

On s'interesse maintenant á la courbe fractale du dragon.
Ecrire une fonction `dragon` qui trace la courbe définie par :

- La courbe d'ordre 0 est un vecteur entre 2 points quelconques P et Q.
- La courbe d'ordre n est la courbe d'ordre $n-1$ entre P et R suivie de la même courbe d'ordre $n-1$ entre Q et R, où PRQ est le triangle isocèle rectangle en R.

Un peu d'aide : Si P et Q sont les points du segment de départ alors :

$$R = P + \overrightarrow{PQ}/2 + \overrightarrow{PQ}^{\perp}$$
$$\overrightarrow{(a,b)}^{\perp} = \overrightarrow{(-b,a)} \textbf{ or } \overrightarrow{(b,-a)}$$

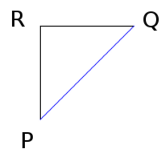
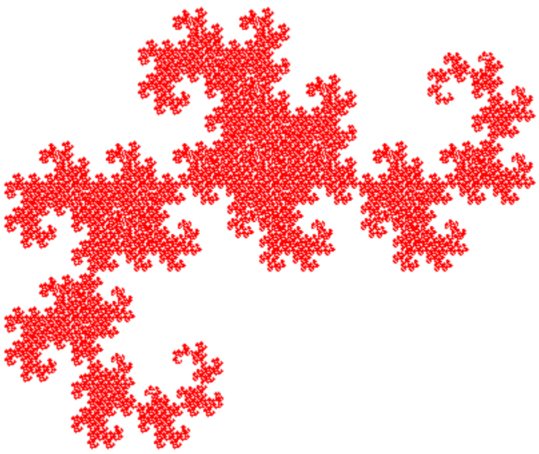


FIGURE 2 – Dragon rang 1

Exemple d'application pour obtenir un joli dragon :

$clear\_graph();;$
$dragon(150,150)(350,350)(19);;$



5 Flocon de Koch

Le flocon de Koch est défini par :

— Le flocon d'ordre 0 est un triangle équilatéral.

— Le flocon d'ordre 1 est ce même triangle dont les côtés sont découpés en trois et sur chacun desquels s'appuient un autre triangle équilatéral au milieu.

— Le flocon d'ordre n consiste á prendre un flocon dórdre $n-1$ en appliquant la même opération sur chacun de ses cotés.

En terme de motifs, le flocon peut s'expliquer ainsi : il est constitué de trois courbes de Koch placées sur les côtés dún triangle équilatéral. La courbe de Koch se construit de la manière suivante : un segment de longueur d sera remplacé par 4 segments de longueur $d/3$, lángle des segments obliques est 60 degrés.

FIGURE 3 – Courbe de Koch ordre 1

Ecrire tout d'abord une fonction rotation qui prend en argument un vecteur $\overrightarrow{v}$ (`int*int`) ainsi qu'un angle θ en radiant (**float**) et qui renvoit le vecteur $\overrightarrow{u}$ representant $\overrightarrow{v}$ auquel on a appliqué une rotation d'angle θ.

Rappel :

$$\overrightarrow{u} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \overrightarrow{v} \tag{2}$$

Ecrire ensuite une fonction récursive qui trace la courbe de Koch **courbeKoch** á partir dún segment et d'un entier n représentant le rang de la courbe. On utilisera la fonction rotation implementée précédemment

FIGURE 4 – Courbe de Koch ordre 2

Enfin, écrire la fonction **Koch** qui trace un flocon complet au rang n donné en utilisant la fonction rotation pour un triangle equilateral de depart sur lequel on applique la fonction **courbeKoch** sur chacun de ces côtés.

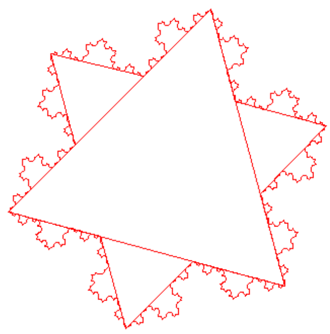


FIGURE 5 – Flocon de Koch ordre 5