UNIVERSITÉ LIBRE DE BRUXELLES

Informatique (INFO-F206) – Projet

Jean Cardinal

Octobre 2021

Fractales

Le but de ce projet est de mettre en œuvre et visualiser un processus itératif permettant de dessiner des objets dits fractals. Ces objets mathématiques ont la particularité d'être composés de parties similaires, et chacune de ces parties est elle-même semblable à l'objet lui-même. On parle d'auto-similarité. Un exemple classique d'occurrence de cette propriété dans la nature est la feuille de fougère. Une feuille est composée de plusieurs sous-feuilles, chacune identique à la feuille elle-même. D'autres végétaux, comme le chou romanesco, ont également une structure auto-similaire. Une illustration est donnée ci-dessus. Les objets auto-similaires, et les fractales en général, ont été popularisés auprès du grand public depuis les années 1980. Ils sont en particulier utilisés en image de synthèse et apparaissent naturellement dans de nombreux domaines des mathématiques et des sciences naturelles.

Le jeu du chaos

Le processus que nous allons décrire permet de visualiser facilement une large famille de tels objets. On parle de systèmes de fonctions itérées ("iterated function system", ou IFS), mais le processus porte aussi le nom de Jeu du chaos.

Nous notons (x, y) les coordonnées d'un point dans le plan. Nous disposons en guise de description de notre objet une collection de k transformations affines. La transformation d'indice i (pour i compris entre 0 et $k-1$) est définie par six coefficients notés a_i, b_i, c_i, d_i, e_i, f_i. Elle transforme le point (x, y) en un point (x', y') tel que :

$$\begin{aligned} x' &= a_i \cdot x + b_i \cdot y + c_i \\ y' &= d_i \cdot x + e_i \cdot y + f_i. \end{aligned}$$

L'algorithme permettant de dessiner l'objet correspondant à cette collection de transformations est le suivant :

1. Initialiser x et y à 0.
2. Répéter n fois :
   (a) Choisir un entier aléatoire i uniformément distribué entre 0 et $k-1$ (inclus)
   (b) Remplacer x et y par x' et y', leurs images par la $i$ème transformation
   (c) Afficher le point de coordonnées (x, y)

L'objet affiché sera donc constitué d'un nuage de n points, défini par les valeurs successives prises par la paire de variables x, y. Ces valeurs sont déterminées par un processus aléatoire, puisque l'algorithme sélectionne itérativement une des k transformations au hasard. Le nombre n est choisi suffisamment grand, de façon que la répartition des points dans le plan représente l'objet de manière effective.

Le dragon

Un exemple simple de tel objet est appelé la courbe du *Dragon*. Il s'agit du point fixe du processus décrit ci-dessus pour les deux transformations suivantes :

$$\begin{aligned} x' &= x/2 - y/2 \\ y' &= x/2 + y/2, \end{aligned}$$

et

$$\begin{aligned} x' &= x/2 - y/2 + 1 \\ y' &= x/2 + y/2. \end{aligned}$$

Travail demandé

Nous vous demandons de concevoir un programme complet en Python 3 permettant de calculer et sauvegarder une représentation d'un objet fractal défini par un système de fonctions itérées. Les transformations seront lues dans un fichier. Le nom de ce fichier sera passé en paramètre au programme. Un second paramètre passé au programme sera le nombre n d'itérations à effectuer.

Le contenu du fichier sera au format suivant :
- la première ligne contiendra le nombre k de transformations,
- les coefficients de chaque transformation i seront donnés sur une ligne dans l'ordre suivant : a_i, b_i, c_i, d_i, e_i, et f_i. Sur chaque ligne, les coefficients sont séparés par au moins un espace.

Le programme créera un fichier nommé `out.txt` qui contiendra la liste des coordonnées x, y successives obtenues avec le processus. Les coordonnées seront séparées par un espace, et on écrira exactement une paire de coordonnées par ligne du fichier.

Pour cela, vous devrez utiliser les structures du langage Python vues en cours. En particulier, on vous demande :

- de diviser le programme en fonctions élémentaires,
- de commenter de manière pertinente les différentes parties du programme (notamment en utilisant les "docstrings", voir syllabus, Chapitre 3, "Documenter ses fonctions"),
- d'utiliser le générateur de nombres pseudo-aléatoires, les fonctions de manipulation de fichiers et le passage de paramètres au programme, comme décrit ci-après.

Nous mettons à votre disposition un petit utilitaire, écrit également en Python, permettant de visualiser les points générés par votre programme.

Consignes

On vous demande un programme Python complet et dûment commenté sous forme d'un fichier avec l'extension `.py`. Le travail est à rendre sur l'Université Virtuelle pour le **20 décembre 2021**.

Informations utiles

Passage de paramètres au programme

Il est possible de passer des paramètres au programme principal. Pour cela, dans l'interpréteur Thonny, il faut sélectionner l'option *Affichage* → *Arguments du programme*. Une nouvelle boîte apparaît, dans laquelle on peut taper du texte. Chacun des mots de ce texte sera un paramètre distinct passé au programme.

Pour accéder à ces paramètres dans le programme, celui-ci doit avoir la commande suivante dans son en-tête :

```
import sys
```

Les différents paramètres sont ensuite accessibles via les variables `sys.argv[1]`, `sys.argv[2]`, etc. Ces variables sont toutes des chaînes de caractères, donc de type `str`.

Lecture et écriture dans un fichier

La fonction `open(nom)` prend en paramètre un nom de fichier de type `str` et renvoie un objet qui permet de le lire. Pour lire le fichier `fougere.txt`, on écrira par exemple :

```
fichier = open ('fougere.txt')
```

La fonction `fichier.readline()`, appelée sur l'objet `fichier`, renvoie une chaîne de caractères contenant la ligne suivante dans le fichier. On peut de plus parcourir toutes les lignes du fichier en utilisant une boucle `for` comme suit :

```
for ligne in fichier:
    # la variable ligne de type str contient la ligne suivante du fichier
```

Lorsque la lecture est terminée, on ferme le fichier en utilisant `close()` :

```
fichier.close()
```

Pour écrire dans un fichier, il faut d'abord l'ouvrir en mode "écriture" (write) :

```
fichier = open ('out.txt', 'w')
```

On peut ensuite y écrire ligne par ligne en utilisant la fonction `write`, qui prend une chaîne de caractères en paramètre :

```
fichier.write('bonjour !')
```

Lorsque l'écriture est terminée, on ferme le fichier en utilisant `close()` :

```
fichier.close()
```

Génération de nombres pseudo-aléatoires

Le module `random` contient un certain nombre de fonctions utiles pour générer des nombres qui ont toutes les apparences de nombres aléatoires. En particulier, la fonction `random.randint(a,b)` renvoie un entier pseudo-aléatoire compris entre `a` et `b` inclus.

Pour accéder à ces fonctions dans le programme, celui-ci doit avoir la commande suivante dans son en-tête :

```
import random
```

Visualisation

Nous vous fournissons un petit programme Python `plot.py` permettant de visualiser le contenu du fichier `out.txt` sous forme d'un nuage de points. Il utilise les fonctions du module `graphics.py`, fourni également. Le fichier `out.txt` à visualiser doit se trouver dans le même répertoire que ces deux fichiers Python.

Exemples de fichiers

Voici les coefficients des transformations pour la fougère fractale, au format prescrit :

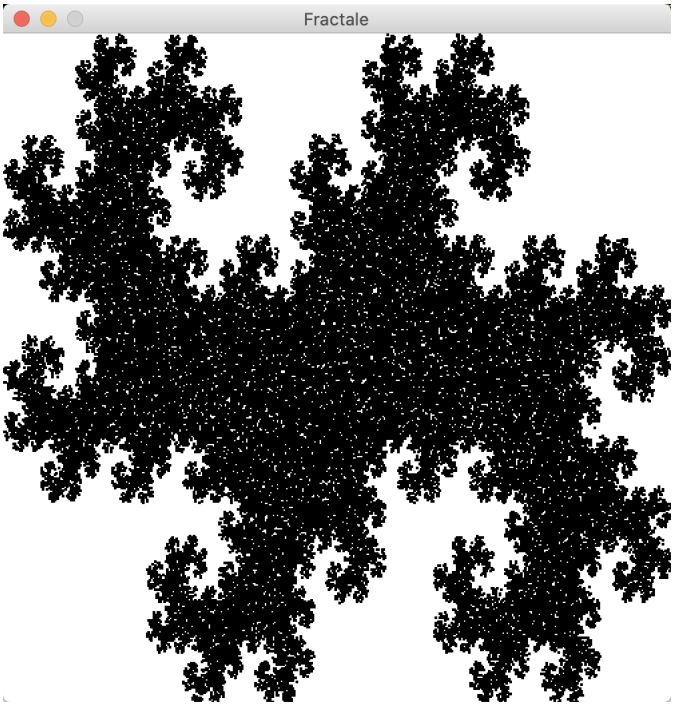
```
12
0.0 0.0 0.0 0.0 0.16 .0
.85 .04 0.0 -.04 .85 1.6
.85 .04 0.0 -.04 .85 1.6
.85 .04 0.0 -.04 .85 1.6
.85 .04 0.0 -.04 .85 1.6
.85 .04 0.0 -.04 .85 1.6
.85 .04 0.0 -.04 .85 1.6
.85 .04 0.0 -.04 .85 1.6
.85 .04 0.0 -.04 .85 1.6
.85 .04 0.0 -.04 .85 1.6
.2 -.26 0.0 .23 .22 1.6
-.15 .28 0.0 .26 .24 .44
```

Et le résultat obtenu pour $n = 100000$ itérations :

Les coefficients pour la courbe du dragon sont :

```
2
0.5 -0.5 0.0 0.5 0.5 0.0
0.5 -0.5 1.0 0.5 0.5 0.0
```

Et le résultat obtenu pour $n = 100000$ itérations :



Pour en savoir plus

Des vidéos du projet MITK12 illustrent de manière simple les concept liés aux objets fractals, voir par exemple ici et là.

Un des premiers articles concernant les systèmes de fonction itérées décrits ici a été publié par Hutchinson en 1981 : *John Hutchinson, « Fractals and self similarity », Indiana University Mathematical Journal, 1981, p.714.* Il est accessible en ligne à cette adresse.

La page Wikipedia consacrée aux systèmes de fonctions itérées renvoie à d'autres références pertinentes.