# Programmation temps-réel, Projet

Aline Huf. <alinehuf@ai.univ-paris8.fr>

2016-2017

## 1 Consignes générales

Réalisation du projet :

- Les différentes version du programme doivent être réalisées en langage C ou C++, utiliser la bibliothèque OpenCV (OpenGL interdit) et tourner sous Linux.
- Privilégiez un style clair et lisible
- Nommez les fonctions et les variables avec des noms appropriés
- Commentez votre code
- Soignez votre indentation
- **Chaque partie doit faire l'objet d'un programme séparé.**
- **Les documents accompagnant de projet** (présentation, graphique) **doivent être fournis au format PDF** pour s'assurer que la mise en page soit préservée (et que le correcteur puisse les ouvrir).
- Si votre programme comporte plusieurs fichiers, regroupez-les dans un dossier.

Remise du projet :

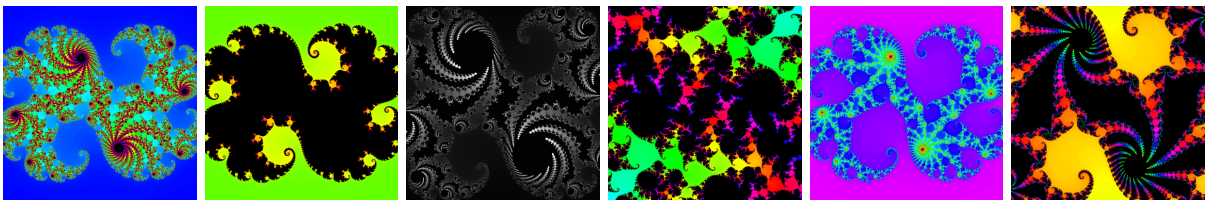
- Rendre une archive .tar.gz contenant d'un répertoire nommé PROJET.
- Nommer l'archive par vos NOM, PRENOM et NUMERO (sur le modèle NOM1-PRENOM1-NUMERO1_NOM2-PRENOM2-NUMERO2.tgz).
- Le répertoire de l'archive contient les différentes versions du programme (un seul fichier ou un dossier par programme).
- Chaque fichier ou dossier doit être nommé avec le numéro de la version (ex : "partie1" ou "etape1"). Déposer l'archive sur : https://moodle.univ-paris8.fr/moodle/course/view.php?id=1373
- Un seul membre du binôme doit déposer le devoir, mais faites attention à bien préciser le nom de l'autre participant.

**Date limite de dépôt du projet : le 4 décembre 2016 à 12h.**

## 2 Présentation du projet

Vous allez utiliser des threads POSIX pour faire un calcul parallèle de fractales de Julia (Chaque pixel de l'image peut se calculer indépendamment des autres). Le thread principal (processus parent) se chargera de l'affichage et de l'interaction avec l'utilisateur. Les threads secondaires réaliseront le calcul de la fractale et modifieront les pixels de l'image.

Voici un exemple des images qui pourront être obtenues :



L'utilisateur pourra en tapant sur différentes touches du clavier : modifier la structure de la fractale, sa couleur, zoomer/dé-zoomer ou demander à quitter l'application. La fractale sera modifiée et ré-affichée à la volée.

Chacune des étapes de la conception du programme (indiquées ci-après) devra être présentée ainsi qu'une étude du temps nécessaire pour le calcul de la fractale. Vous utiliserez OpenCV pour afficher la fractale. Les fichiers `opencv_exempleC.cpp` et `opencv_exempleCPP.cpp` donnent un exemple minimal de l'utilisation d'openCV en C et en C++ avec toutes les fonctionnalités dont vous aurez besoin. Les deux exemples sont compilés avec g++ pour plus de simplicité. Pour compiler avec gcc, il est conseillé d'utiliser cmake (voir la documentation).

# 3 Première étape : Calcul de la Fractale de Julia

Étant donnée une constante complexe $c$, on définit la suite complexe $z_{n+1} = z_n^2 + c$. Chaque constante $c$ correspond à une image différente (la fractale aura une structure différente). Á chaque point de l'image de coordonnées $(x', y')$, on associe le nombre complexe $z_0 = x + i.y$ qui permet d'initialiser la suite. On peut alors calculer les termes de la suite et étudier son comportement. Si la suite est bornée alors elle fait partie de l'ensemble de Julia et on mettra un pixel noir. On décidera que la suite est bornée si au bout d'un nombre d'itérations $N$, le module de la suite n'a pas dépassé une certaine valeur $V$. Dans le cas contraire, on mettra un pixel plus ou moins blanc en fonction du nombre d'itérations effectué avant de dépasser la valeur $V$.

Les images présentées en exemple on été obtenues pour les valeurs $N = 300, V = 4, x \in [-1, 1]$ et $y \in [-1, 1]$. Si votre image a une taille de $1024 \times 1024$ pixels, il faut donc transposer les coordonnées $(x', y')$ de chaque pixel pour obtenir la valeur $z_0 = x + i.y$ correspondante.

Écrivez une première version du programme qui calcule la fractale de manière séquentielle (sans threads). Vous aurez besoin d'une fonction pour calculer la valeur d'un pixel en fonction de $x'$, $y'$ et $c$. Ensuite vous pourrez utiliser cette fonction pour calculer chaque pixel de l'image.

Pour affichez une fractale en couleurs, faites une petite recherche sur internet pour trouver une manière de la colorer (indice : on peut passer de HSV en RGB...).

# 4 Deuxième étape : Parallélisation de la Fractale de Julia

Écrivez une seconde version de votre programme qui utilise plusieurs threads pour calculer la fractale. Chaque thread calculera une partie des points.

Mettez au point un banc de test pour évaluer le temps de calcul de la fractale (temps moyen ou temps dans le pire des cas, à vous de voir). Prenez en compte le fait que le temps de calcul varie en fonction du nombre d'itérations effectuées sur chaque point (les différentes versions de la fractale sont plus ou moins longues à calculer) et en fonction du nombre de threads. Nous partirons du postulat que la taille de l'image est fixe.

Pour faciliter les tests, votre programme pourra prendre sur la ligne de commande des arguments correspondants aux valeurs pouvant faire varier le temps d'exécution et que vous aurez décidé de tester. Un script shell (ou python, ou autre) sera utilisé pour lancer le programme avec différentes valeurs.

Vous écrirez un petit texte pour présenter votre protocole de test : qu'est-ce que vous testez, combien de fois vous itérez, sur quelles valeurs, sur quel matériel (performances du processeur, taille de la RAM), quels choix vous avez fait compte tenu que vous n'avez pas énormément de temps pour faire les tests. Vous présenterez les résultats du banc de test sous forme d'un graphique présentant la moyenne des temps d'exécution obtenus pour chaque valeur testée avec une indication de l'écart type.

Vous ajouterez une critique de vos résultats et indiquerez ce que vous pourriez améliorer/tester en plus si vous aviez d'avantage de temps.

# 5 Troisième étape : Interaction avec l'utilisateur

Écrivez une troisième version de votre programme qui permet l'interaction avec l'utilisateur. La fractale doit pouvoir être modifiée à la volée en augmentant ou diminuant la partie réelle ou imaginaire de la constante $c$. L'utilisateur doit également pouvoir zoomer/dé-zoomer ou modifier la coloration de la fractale.