AiFractals WhiteBox

par Adrabi Abderrahim

2010-02-26

0) Introduction.
1) Théorie.
2) Complexité et Problème.
3) Solution, Problème de « *Solution* », Solution de *« Problème de « Solution » »*
4) Buddhabrot.
5) Conclusion.

0) Introduction.

AiFractals, est créer pour être dynamique et facile a utiliser, avec l'utilisation des déférents modules de Framework Qt, et lui aussi se devise par deux grands partie :

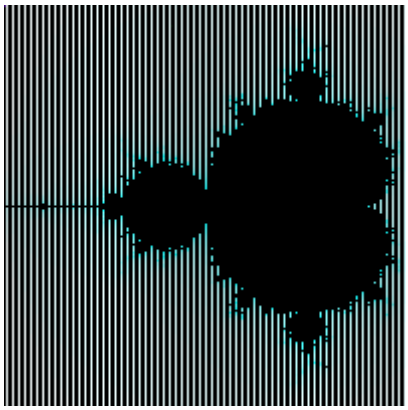
- La partie GUI (statique il ne change pas que avec le changement de son code source).
- La partie dynamique qui est dédie pour les utilisateurs d'AiFractals.

Pour ça soit possible, AiFractals se base sur une théorie à lui seul, pour qu'il puisse adopter tous les changements externes (les scripts).
(Je ne vais pas écrire des codes source ici, merci de les vois dans code source)

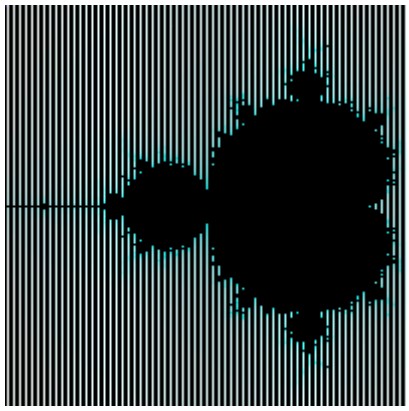
1) Théorie.

Théorie d'AiFractals, se base sur le nombre de « Threads » **définir** par utilisateur pour calculer automatiquement *les points de débuts* et *les points d'incrémentations*, pour générer des images unique, on dit que chaque « Thread génère seulement une unique image de la zone complexe », si il y a plus « une », alors la théorie est fausse si seulement si l'algorithme est correct, le problème comment savoir si l'algorithme est incorrect ?

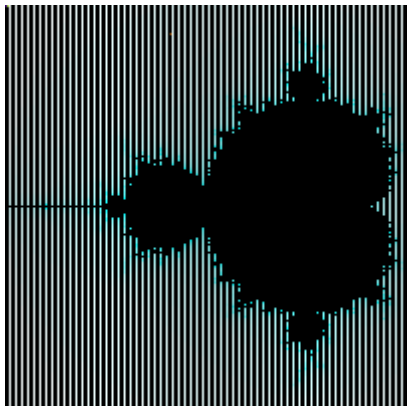
Après la génération (ou création) des images de chaque « thread » avec l'utilisation des points de débuts et d'incrémentations, il va donne des images avec des trous transparent ou noir, pour les accumuler est donne une seule unique image avec considération que les tours de chaque image ce sont des places **réserve** (Pixels) pour les autres Pixels des images de la série (les autres images générée par les autres threads), exemple:



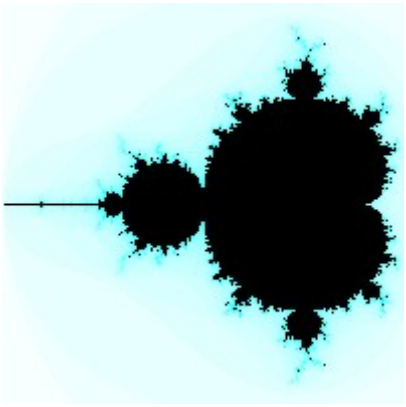
Thread 1



Thread 2



Thread 3

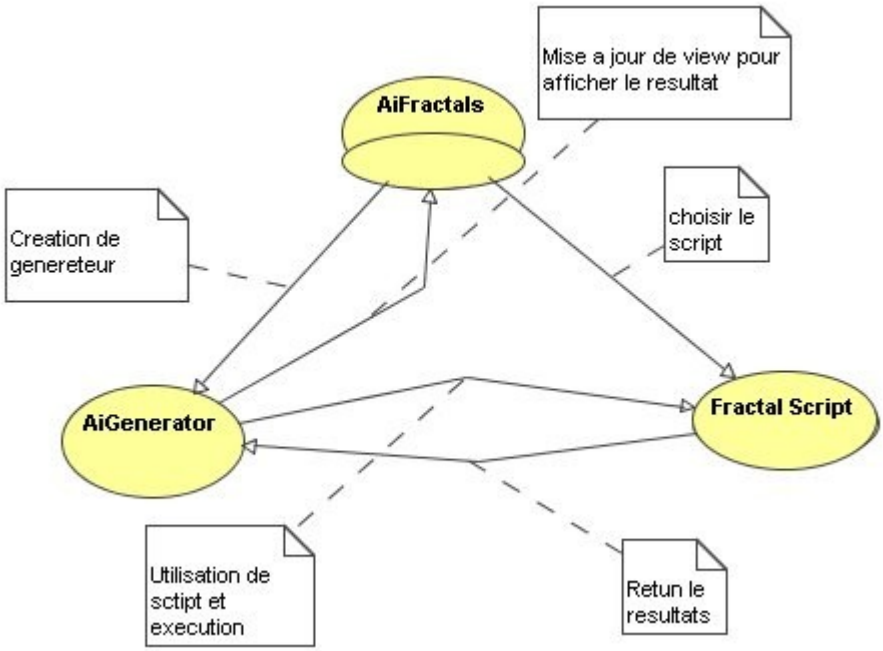


Résultats Thread 1 + Thread 2 + Thread 3

2) Complexité et Problème

Complexité se trouver comment mettre tous ça on marche, et la réduire sans créer un horrible source code, pour tout ça, AiFractals a créer ça propre architecture il presque comme MVC :

- Le générateur, AiGenerator (C)
- L'interface graphique, AiFractals (V)
- Les scripts, Fractal Script (M)



AiGenerator, c'est le générateur de Fractals, il se compose de deux classes seulement AiThread et AiZone.
AiThread, c'est la classe responsable sur la génération, elle utilise AiZone pour crée une zone complexe parfait, pour qu'elle aussi créer une image de thread.

AiFractals, c'est l'interface utilisateur statique pour enter les chiffres et d'autres manipulations, ce compose de multi-dossier pour la simplification (source code).

Fractal Script, c'est la partie dynamique qui se base sur JavaScript avec un objet spécial « zone ».

Problème, ou les problèmes : « qui » va contrôler « qui » ?, comment faire les calcules et lancer threads ?, avec « quoi » la partie dynamique est contrôlé ?, comment créer un générateur avec plusieurs threads sans bloquer ou avoir des problèmes avec GUI et les manipules ?, comment manipule les résultats ?, etc...

3) Solution, Problème de « *Solution* », Solution de « *Problème de « Solution » »*

* Solution *

La solution c'est de créer un générateur qui ce compose de multi-générateurs, le générateur est créer avec équation :

Générateur = N * AiThread et N > 0 et N <= 99

Et chaque AiThread contaient seulement un unique AiZone, qu'on la génération est terminer AiThread **émit un signal** avec l'objet AiZone a AiFractals (AiMainWindow), pour faire l'accumulation des *images thread* et donne une image complète, avec l'utilisation de composition modes définir par utilisateur.

* Problème de « *Solution* » *

Le problème, on théorie tous threads sont lancer dans le même temps (une déférence de 1~10nanos), qui va créer un désordre des images thread au moment d'accumulation avec composition modes (voir GUI 1er composition mode et autres composition mode), pourquoi ? Parce que dans la théorie on dit « image unique qui utilise un point de début et des points d'incrémentation » alors les derniers threads lancer se sont aux les premier terminer !

Exemple :
On a 3 threads,

Le 1 AiThread il à un point de début = 0 ; points d'incrémentation = 3
Le 2 AiThread il à un point de début = 1 ; points d'incrémentation = 3
Le 3 AiThread il à un point de début = 2 ; points d'incrémentation = 3

Alors le 3 AiThread, c'est lui le plus proche a la fin de la limite (dimensions de l'image), alors c'est lui le 1er qui va terminer.

* Solution de *« Problème de « Solution » »* *

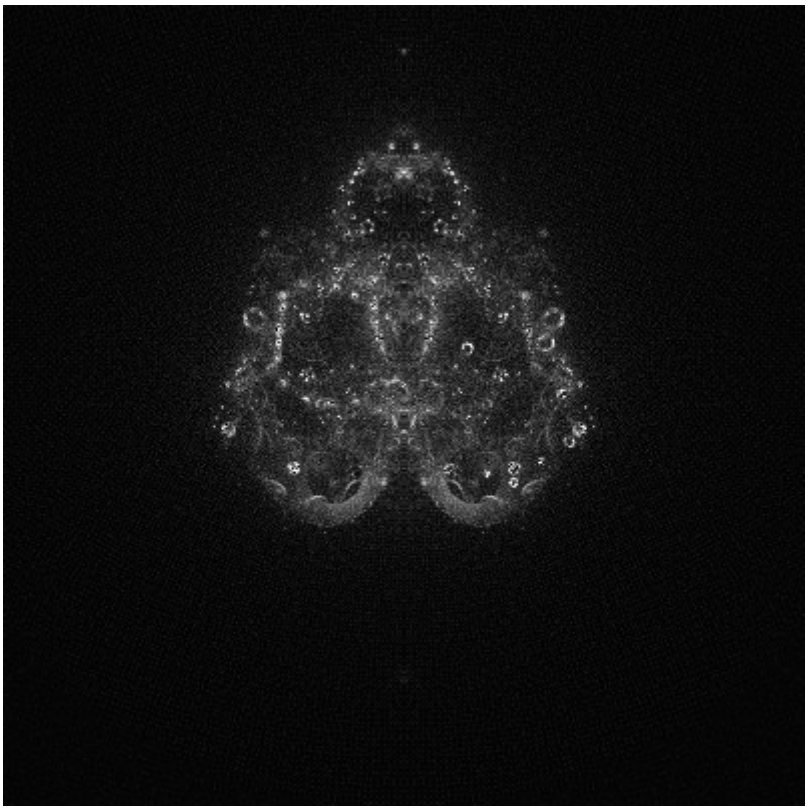
La solution est simple, chaque AiThread possède un ID unique dans son groupe de générateur, et cet ID c'est son point de début, et pour cela la classe AiMainWindow possède un espace de stockage (QVector m_storage) réserve au moment de début de la création de générateur, pour inséré ou remplace l'espace réserve par AiZone de chaque AiThread, et qu'on tous AiThreads sont terminer, l'accumulation commence avec l'ordre croissant ou le l'ordre de création de AiThreads.

4) Buddhabrot

Après tous ça, on peut dire -- que sa rien avoir avec Buddhabrot -- parce que, pour la génération de Buddhabrot il faut utilise des points aléatoires, et la théorie ce base uniquement sur les point **statique** et **définie**, et c'est pour ça j'ai du créer une nouvelle méthode de génération de Buddhabrot, qui adopte tous ce qu'il est dit dans ce document, le script c'est :

Buddhabrot-v1.js (avec colorisation)
Buddhabrot-v3.js (sans colorisation)

Pour donne la vie a :



5) Conclusion

On réalité AiFractals ne possède pas de complexité des algorithmes, ou les composants, seulement la manière de faire et prévoir les bugs ou les mauvaises manipulations, qui peuvent bugée AiGenerator.

AiGenerator, c'est une partie de AiFractals, il est séparer de lui pour qu'il soit utiliser dans d'autres applications ou concept « Network rendering » par exemple, ou des simples de-serialisateurs c'est on parle de God Mode.

God Mode, si seulement la sérialisation de générateur (N * AiThread), est la sauvegarde de ses états (le nombre de threads, les dimensions, les itérations, zoom, etc. …) pour être indépendant de GUI.

GUI, il offert plusieurs fonctionnalité, comme zooming, sauvegarde des images, rotations, création de fractals personnalise avec utilisation d'autre images etc... plus il a un debugger des scripts pour pouvoir savoir c'est l'algorithme est incorrect.

Pour un seul but, c'est de créer le plus possible un générateur de Fractals qui fait ça place dans le monde Open Source et les autres générateurs des autres participants.