HLIN405
Projet de Programmation
Visualisation et Exploration de l'ensemble de Mandelbrot

Melvin BARDIN, Alexandre CANTON CONDES,
Ambre LAMOUCHI, Malika LIN-WEE-KUAN

UNIVERSITÉ DE MONTPELLIER

FACULTÉ DES SCIENCES MONTPELLIER

Table des matières

Lancement du programme

Les différentes fenêtres

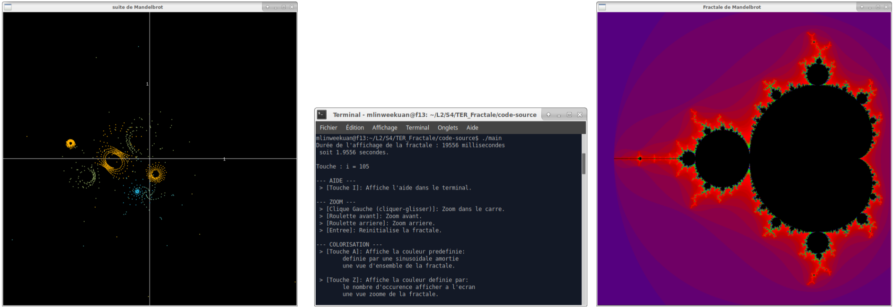
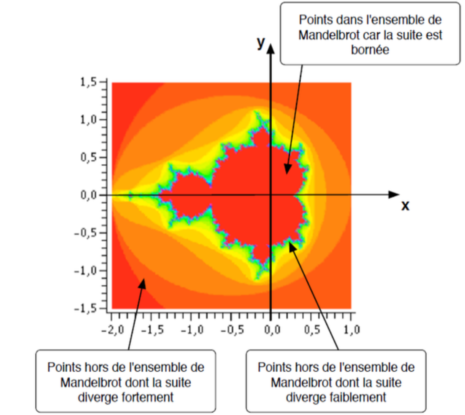


Figure – Les fenêtres : Suites, Terminal, Mandelbrot/Julia

Rappels sur les fractales

L'ensemble de Mandelbrot

Mandelbrot



L'ensemble de Mandelbrot :

$$\begin{cases} z_0 = 0 \\ z_{n+1} = z_n^2 + c \end{cases}$$

Rappels sur les fractales

Les fractales de Julia

Julia

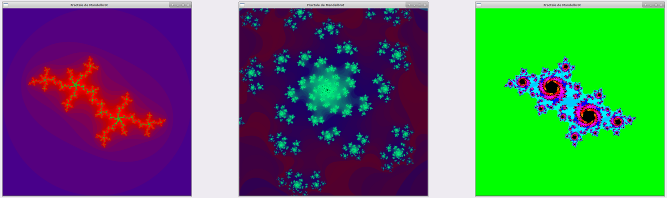


Figure – Exemples de Fractales de Julia

On a la formule $z_{n+1} = z_n^2 + k$ avec $Z_0 = p$ et k constant aux différents points p du plan complexe, on fait la distinction entre les points qui génèrent des suites convergentes et divergentes.

Architecture du code

OpenGL

OpenGL : une librairie graphique

Composée d'un panel de fonctions d'affichage en 2D ou 3D.

GLU et GLUT : des extentions

Permettant de gérer les fenêtres et les évènements.

Un langage à état

Signatures de fonctions strictes ne permettant pas de passer des paramètres supplémentaires. Obligation d'utiliser des variables globales pour le bon fonctionnement du programme.

Figure – Logo OpenGL

Architecture du code

Explication des fonctionnalitées du programme

Organisation du projet

Un fichier contenant la fonction "main".

Deux classes.

Deux fichiers regroupant les diverses fonctions utilisées.

Lors de l'execution du programme

Deux fenêtres s'ouvrent :

- L'ensemble de Mandelbrot
- Les suites de mandelbrot

Possibilité de passer de Mandelbrot aux fractales de Julia et vice versa.

Architecture du code

Colorisations

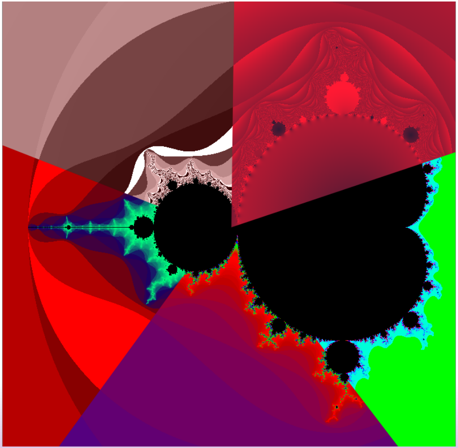


Figure – Les différentes colorisations

Architecture du code

Pseudo-code de la fonction Diverge

Algorithme 3.1: diverge(r, i)

Commentaire: r, i respectivement partie réelles et imaginaires du complexe.

Commentaire: Zn est un couple (R,I) avec R la partie réelle et I la partie imaginaire.

Commentaire: on nommera Zn(R) sa partie réelle et Zn(I) sa partie imagniaire

$Zn \leftarrow (0, 0)$
$compteur \leftarrow 0$
tant que $compteur < 100$ **et** $module(Zn) < 2$
faire
$\begin{cases} Zn \leftarrow (Zn(R)^2 - Zn(I)^2 + r, 2 * Zn(R) * Zn(I) + i) \\ compteur \leftarrow compteur + 1; \end{cases}$
si $compteur == 100$
$\{$**retourne** (-1)
sinon
$\{$**retourne** $(compteur)$

Conclusion

Quelques chiffres

Développement

- **Temps de développement :**
  - 3 mois et demi de développement
  - 5 heures et demi de travail par semaine
- **GitLab :**
  - 199 commits répartis sur 107 jours
  - 1.9 commits par jour en moyenne
- **Projet final :**
  - 4 fichiers d'en-tête
  - 4 fichiers de code source
  - 1 fichier de code principal *(contenant la fonction main)*
  - 1 Makefile
  - 1630 lignes de code *(sans compter le README)*

Conclusion

Bilan

Bilan

On a vu :

- L'ensemble de Mandelbrot
- Les fractales de Julia
- La programmation graphique
- L'utilité de la colorisation dans l'analyse de fractales
- Le travail en équipe

Conclusion

Ouverture : Optimisation

Temps de calcul

Ne pas recalculer les suites lors d'un changement de repère.

Colorisation

Ajouter d'autres méthodes de coloration pour un nouveau point de vue de la fractale.

Conclusion

Démonstration