# Sommaire :

1. Résultats attendus
2. Démarche générale
3. CUDA : création du programme de calcul avec plusieurs GPU
4. PYTHON 1 : Création des images taille réel avec plusieurs GPU
5. PYTHON 2 : Création DZI
6. WEB : Création site WEB
7. Remerciements

# Résultats Attendus

On chercher à obtenir des fractale de Julia en deux formats d’image avec une taille importante (>32k px de coté) :

* Bi couleur (N&B)
* En nuance de gris

Une image contenant Graphique, art, symbole, cercle

Description générée automatiquement

Figure : Image bi-couleurs de la fractale de Julia

Une image contenant ciel, obscurité, nuage, noir

Description générée automatiquement

Figure : Image en nuance de gris de la fractale de Julia

De même on, utilisera un site web locale ou sur réseaux pour visualiser les fractales de Julia, l’interface se tel que :

Une image contenant texte, capture d’écran

Description générée automatiquement

Figure : Interface WEB

Description de l’interface WEB :

1. Titre Dynamique avec les valeurs de X et Y
2. Axe des X : permet de modifier la valeur de X
3. Axe des Y : permet de modifier la valeur de Y
4. Option d’affichage et bouton de téléchargement de l’image d’origine
5. Explorateur de la fractale, avec un zoom important possible.

# Démarche générale

Il y a 4 étapes à respecte :

1. Création d’un tableau du nombre d’itération de chaque pixel de l’image
   * + Outil : CUDA (C / C++)
     + OS : Linux (WSL 2 Ubuntu)
     + Matériel : Carte graphique Nvdia 4 Go RAM
2. Transformation du tableau du nombre d’itération en images et compression du tableau pour optimise l’usage du disque dur.
   * + Outil : CUDA (C / C++) et python 3
     + OS : Linux (WSL 2 Ubuntu)
     + Matériel : Carte graphique Nvdia 4 Go RAM
3. Création d’image zoomable avec le logiciel « openseadragon » et « deepzoom.py »
   * + Outil : python 3
     + OS : Linux (WSL 2 Ubuntu) ou Windows
4. Création du site web pour visualiser les fractales
   * + Outil : python 3 / HTML / JS
     + OS : Linux (WSL 2 Ubuntu) ou Windows
5. CUDA : création du programme de calcul avec plusieurs GPU

Le code cuda permet d’utiliser les GPU NVDIA comme centre de calculs.

Le code que je propose est décompose en 5 parties :

1. Le header

C’est le code commun entre le code cuda et c++, on y trouve :

* Le type de fractale à générer : Type\_Fractal
* La structure **Complex** pour représenter les nombres complexes
* La structure **ParameterPicture** pour stocker les paramètres de l'image fractale

1. Le code cuda

C’est le code qui calcul la fractale de Julia ou de Mandelbrot, on y trouve :

* Kernel\_Picture : Kernel CUDA pour générer une image fractale
* RUN : la fonction pour exécuter le kernel CUDA

1. Le code C++

C’est le code qui permet de gérer la création de fractales de Julia ou de Mandelbrot, on y trouve :

* File\_Generate : la structure pour gérer les fichiers (.bin et .txt)
* RUN : Déclaration de la fonction CUDA externe
* CreateFolder : Fonction pour créer le dossier de travail
* if\_file\_exist : Fonction pour vérifier si un fichier existe
* write\_bin : Fonction pour écrire des données binaires dans un fichier
* run : Fonction supervision pour lancement de calculs d'une fractale
* Get\_nbfiles\_bin : Fonction pour obtenir le nombre de fichiers binaires existants
* Open\_file\_txt : Fonction pour ouvrir un fichier texte et lire son contenu
* Main : Fonction principale qui est exécuté au lancement

1. Le scripte pour compiler le programme.

C’est le scripte qui permet de générer l’application

1. Les paramètres

C’est les paramètres de calculs externes au programme, on y trouve :

* L’id de la care nvdia à utiliser de 0 à N, n étant le nombre -1 de cartes graphiques disponibles
* La borne minimale du coef y de Julia
* La borne maximale du coef y de Julia

1. PYTHON 1 : Création des images taille réel avec plusieurs GPU
2. PYTHON 2 : Création DZI
3. WEB : Création site WEB

# Remerciements