

Rapport SAE Détection de biomes sur des exoplanètes

Fonctionnement

Flou

Afin de détecter nos différents biomes, on commence par appliquer un flou à l'image. L'objectif est de réduire le gradient entre les différents pixels afin d'uniformiser l'image et de faciliter les opérations de nos algorithmes.

Nous avons implémenté deux méthodes de flou :
Le flou par moyenne et le flou gaussien.

Le flou par moyenne est plus simple mais moins précis puisqu'il uniformise trop les pixels et ne prend pas forcément en compte de grosses différences sur certains pixels autour.
Le choix en application s'est donc porté sur le flou gaussien, qui donne un indice d'importance aux pixels environnants.

Afin d'améliorer les performances au maximum, nous avions commencé par réduire la qualité de l'image (le nombre de pixels en hauteur et en largeur), puis ensuite on applique un flou.
Ainsi, on limitait la taille d'une image à une de l'ordre de 750x500 pixels maximum.

Cependant, après optimisation des algorithmes, ceci n'était plus nécessaire et nous ne faisons plus que flouter l'image.

Détection des biomes

Pour détecter les biomes, nous avons choisi l'algorithme KMeans. Il est en effet plus rapide (complexité de n comparé au n^2 de DBScan), et nécessite moins de paramétrage que DBScan. En effet, en fonction de l'image, il faut trouver des paramètres adéquats pour DBScan, ce qui diminue la qualité des biomes obtenus, avec un coût en performance supplémentaire.

Afin de pousser KMeans au maximum de ses capacités, nous avons implémenté un indice de Davies-Bouldin. On calcule ainsi au lancement du programme le nombre de clusters le plus adapté pour obtenir un meilleur résultat avec KMeans.

Voici donc le déroulement de notre algorithme :

- On crée une map associant chaque couleur de l'image au nombre de répétitions de cette même couleur. Cela est utile pour réduire la quantité de calculs à faire et ainsi accélérer l'exécution.
Au lieu de calculer chaque pixel de l'image, le calcul est fait uniquement une fois par couleurs.
- On fait une boucle pour essayer tous les nombres de clusters possibles (la limite est donnée au constructeur)
- On initialise ainsi nos centroïdes aléatoirement
- On crée les groupes de chaque centroïde. Un groupe correspond à des couleurs.
- On recalcule les nouvelles couleurs des centroïdes avec la moyenne des couleurs présentes (calcul avec le nombre d'itérations de chaque couleurs)
- On continue tant que les centroïdes changent (avec un nombre maximum d'itérations)
- Ensuite on calcul l'indice de Davies-Bouldin, quelque peu modifié pour moins privilégier un petit nombre de clusters.
- On conserve le meilleur résultat et on passe à la suite

Nous avons également ajouté une sélection de l'image depuis le terminal pour rendre le programme dynamique.

Etiquetage des biomes

Une fois tous nos clusters obtenus, on peut commencer à afficher la nouvelle image.

Tout d'abord, pour chaque cluster, on uniformise leur couleur par une couleur moyenne du cluster. On peut ainsi parcourir plus rapidement un même cluster par la suite.

La couleur moyenne du biome est ensuite comparée avec la couleur d'un biome la plus proche (Nous avons une map de biomes, qui relie des couleurs rgb à un label).

On génère une carte pour chaque biome trouvé, qui met ce biome en valeur dans l'affichage comparé aux autres.

Exemple :

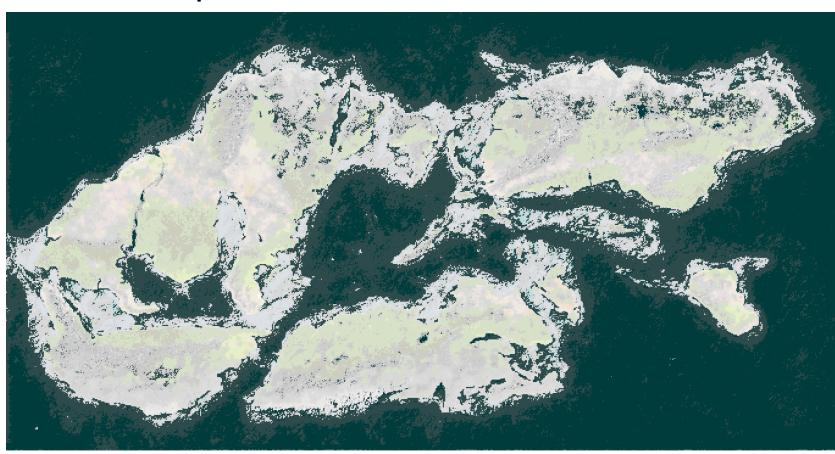
Planète 3 fournie en annexe



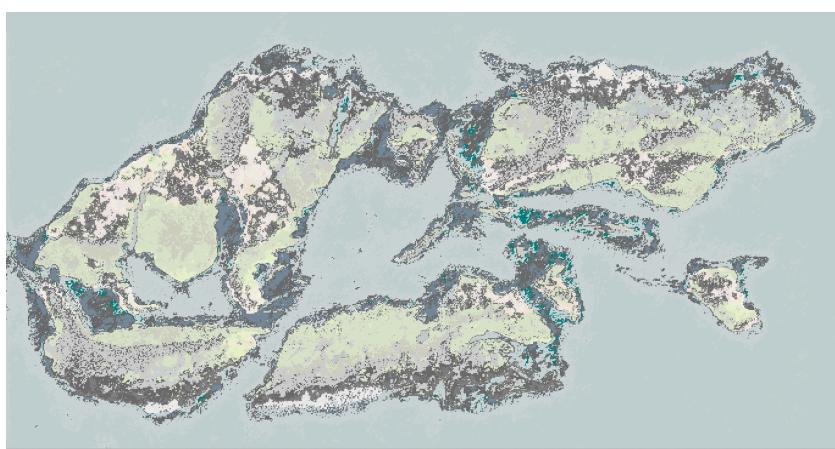
Planete 3_couleur :



Biome Eau profonde



Biome Prairie



Séparation des différentes instances

Pour séparer chaque apparition d'un même écosystème, on fait appel à un autre algorithme : DBScan.

Cette fois-ci, nous avons séparé les milliers (voire millions) de pixels en différent biomes, ce qui rend DBScan plus efficace sur chaque biome comme sa complexité de n^2 est “exponentielle”, dans le sens où si l'on multiplie par 2 le nombre de pixels, le temps d'exécution pourrait être multiplié par 10, 100, ou plus encore comme on traite de gros nombres.

Cette fois-ci, on ne comparera pas les couleurs (rgb) des pixels, mais leur position. Cette modification majeure rend DBScan viable ici car son paramétrage sera beaucoup moins compliqué dans ce cas de figure.

Déroulement de l'algorithme :

- Récupération des pixels d'un biome : Pour chaque cluster identifié comme appartenant à un biome (ex : Prairie), on extrait la liste des coordonnées (x, y) des pixels correspondants.
- Application de DBSCAN : On applique DBSCAN uniquement sur ces coordonnées, avec une distance spatiale eps et un seuil minPts configurables. Ces paramètres indiquent à quel point les pixels doivent être proches pour former une même région.
- Regroupement des écosystèmes : Chaque groupe détecté par DBSCAN est considéré comme une instance indépendante d'un écosystème (par exemple, plusieurs prairies distinctes). Les pixels bruités (isolés) sont ignorés.
- Affichage : Une image est générée pour chaque biome, avec une couleur différente par écosystème détecté, permettant une visualisation claire des zones écologiques séparées.

Exemple :

Planète 2 fournie en annexe



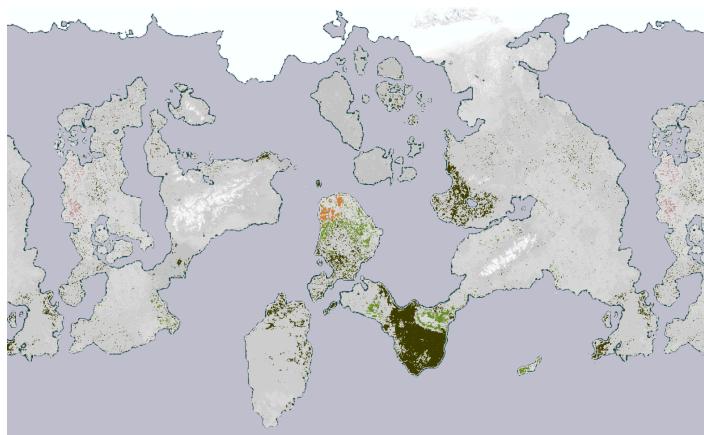
Eau profonde



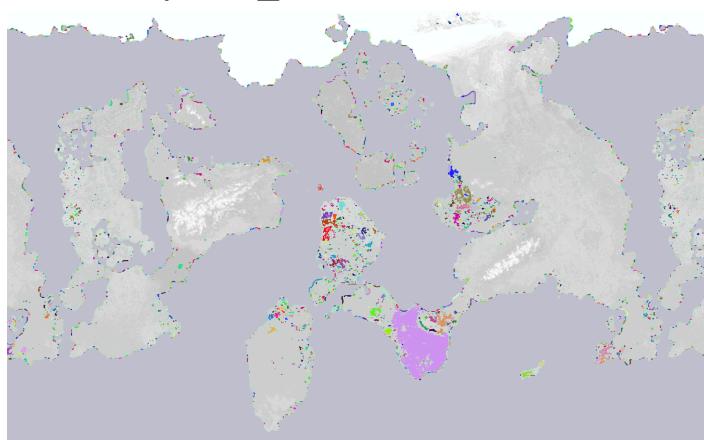
Eau profonde_clusters



Forêt tempérée



Forêt tempérée_clusters



Temps d'exécution

Après être passé à l'oral, nous avons modifié les algorithmes pour les utiliser, notamment en utilisant des Sets au lieu de Listes pour améliorer la vitesse de parcours, ou en augmentant le nombre de points nécessaires pour former un cluster.

Application :

Planete 2 : (2600 x 1600 pixels)

```
C:\Users\elias\.jdks\openjdk-22\bin\java.exe "-javaagent:C:\Program Files\J
Début du calcul des clusters...
Nombre de clusters : 3, Davies-Bouldin : 0.0
Nombre de clusters : 4, Davies-Bouldin : 1574.5764798474534
Nombre de clusters : 5, Davies-Bouldin : 161.4559283431181
Nombre de clusters : 6, Davies-Bouldin : 518.7066193928059
Nombre de clusters : 7, Davies-Bouldin : 207.07317929030782

### RESULTAT ###
Temps d'exécution : 5s

Numéro cluster / Nombre de pixel associé
{0=1626612, 1=186673, 2=2346715}
{0=Prairie, 1=Forêt tempérée, 2=Eau profonde}

Biome Prairie : 1626612
Début du calcul des clusters... (Prairie)
Copie de l'image dans cartes2/Planete 2_Prairie_clusters.png

Biome Forêt tempérée : 186673
Début du calcul des clusters... (Forêt tempérée)
Copie de l'image dans cartes2/Planete 2_Forêt tempérée_clusters.png

Biome Eau profonde : 2346715
Début du calcul des clusters... (Eau profonde)
Copie de l'image dans cartes2/Planete 2_Eau profonde_clusters.png
Temps d'exécution : 6s

Process finished with exit code 0
```

Planète 3 : (2048 x 1082 pixels)

```
C:\Users\elias\.jdks\openjdk-22\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ  
Début du calcul des clusters...  
Nombre de clusters : 3, Davies-Bouldin : 743.540044642492  
Nombre de clusters : 4, Davies-Bouldin : 403.40849808387105  
Nombre de clusters : 5, Davies-Bouldin : 283.19307330668636  
Nombre de clusters : 6, Davies-Bouldin : 494.4281303200726  
Nombre de clusters : 7, Davies-Bouldin : 263.42942139578156  
  
### RESULTAT ###  
Temps d'exécution : 3s  
  
Numéro cluster / Nombre de pixel associé  
{0=26393, 1=22, 2=1490586, 3=58241, 4=14012, 5=616892, 6=9790}  
{0=Désert, 1=Eau peu profonde, 2=Forêt tropicale, 3=Montagne, 4=Désert, 5=Prairie, 6=Désert}  
  
Biome Désert : 26393  
Début du calcul des clusters... (Désert)  
Copie de l'image dans cartes2/Planete 3_Désert_clusters.png  
  
Biome Eau peu profonde : 22  
Début du calcul des clusters... (Eau peu profonde)  
Copie de l'image dans cartes2/Planete 3_Eau peu profonde_clusters.png  
  
Biome Forêt tropicale : 1490586  
Début du calcul des clusters... (Forêt tropicale)  
Copie de l'image dans cartes2/Planete 3_Forêt tropicale_clusters.png  
  
Biome Montagne : 58241  
Début du calcul des clusters... (Montagne)  
Copie de l'image dans cartes2/Planete 3_Montagne_clusters.png  
  
Biome Désert : 14012  
Début du calcul des clusters... (Désert)  
Copie de l'image dans cartes2/Planete 3_Désert_clusters.png  
  
Biome Prairie : 616892  
Début du calcul des clusters... (Prairie)  
Copie de l'image dans cartes2/Planete 3_Prairie_clusters.png
```

```
Biome Désert : 9790  
Début du calcul des clusters... (Désert)  
Copie de l'image dans cartes2/Planete 3_Désert_clusters.png  
Temps d'exécution : 4s
```

```
Process finished with exit code 0
```

On remarquera que le premier temps d'exécution correspond au temps d'exécution de KMeans qui trouve le nombre de biomes adéquat et y regroupe tous les pixels.

Le second temps d'exécution correspond à DBScan qui sépare les instances des biomes et étiquette les biomes en fonction de leur couleur.