Rapport de Projet IA : Détection Automatique de Biomes sur des Exoplanètes

Ryan Korban, Baptiste Hennequin, Baptiste Delaborde, Maxence Eva

Introduction

Le but du projet c’est de développer une détection automatique de biomes et d'écosystèmes sur des images d'exoplanètes. L'objectif est de regrouper les pixels selon leur couleur pour identifier différents biomes (forêt, désert, eau, etc.), puis de détecter les écosystèmes distincts au sein de chaque biome. L'analyse s'est principalement faite sur deux images haute résolution (1400×1400 pixels pour l'image Planète 1 et 2600×1600 pixels pour l'image Planète 2) Mais le projet peut bien sûr analyser n’importe quelle image.

Table des matières

1. Prétraitement de l'Image : Choix du Filtre de Flou

Comparaison des Filtres

Nous avons implémenté les deux types de filtres de flou proposé : le flou par moyenne et le flou gaussien.

Enfaite, le flou gaussien s'est révélé optimal dans notre cas car il permet une préservation des frontières. En gros, il maintient mieux les transitions entre biomes grâce qui en fonction de sa pondération, donne plus d'importance aux pixels centraux. Donc grâce à cela, les transitions entre biomes sont plus naturelles surtout dans notre contexte (des cartes d'exoplanètes) où les frontières entre biomes sont importantes.

Le flou moyen, même s'il est plus rapide, il créée des transitions trop brutes et peut fusionner des biomes distincts qui sont proches.

2. Détection des Biomes

Choix de l'Algorithme : K-Means

Pour la détection des biomes, nous avons choisi l'algorithme K-Means pour trois raisons principales :

- Premièrement, ses performances sont adaptées à nos images de tests (Planète 1 et 2) qui ont respectivement 1,96 et 4,16 millions de pixels. Ainsi grâce à sa complexité en O(n) nous permets d'avoir des résultats rapidement.
- Deuxièmement, la consigne nous indique d'estimer le nombre de biomes ce qui correspond parfaitement au paramètre requis par K-Means.
- Troisièmement, les biomes forment naturellement des zones convexes dans l'espace couleur RGB, ce qui est une configuration idéale pour cet algorithme.

Pourquoi pas DBSCAN ?

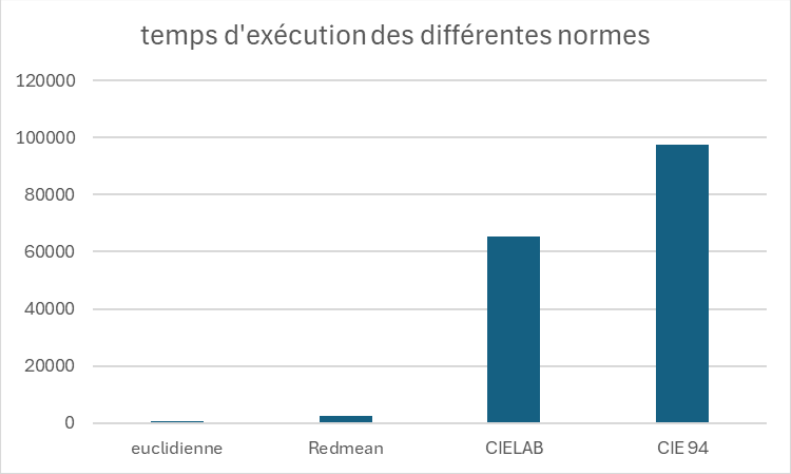
Nous avons écarté DBSCAN car sa complexité en O(n²) devient éliminatoire sur des millions de pixels, et le paramétrage d'epsilon dans l'espace couleur serait trop difficile.

Pourquoi pas HAC ?

L'algorithme HAC a également été rejeté vu que sa complexité est encore plus élevée, il est en O (n² log n).

Choix de la Métrique de Couleur

Grâce au TP de préparation, on avait déjà les Métrique de couleur, pour faire notre choix, on les a toutes testé :

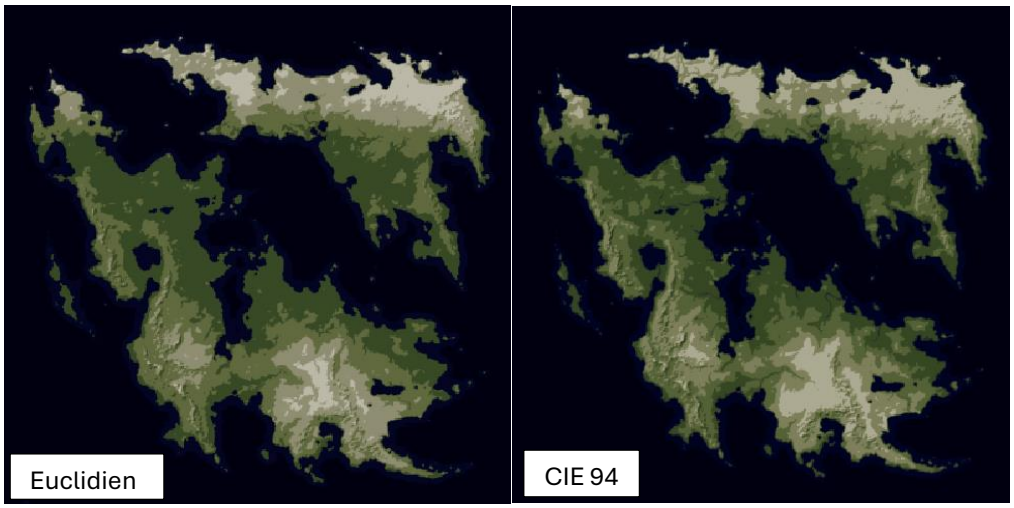


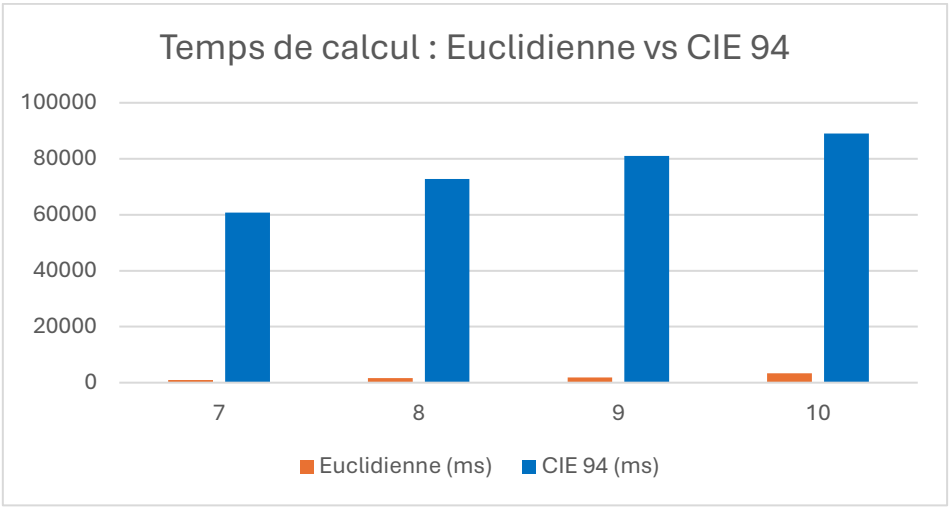
Métrique	Temps (ms)	Qualité visuelle du résultat
Euclidienne	~1300	Correcte
Redmean	~2500	Bonne
CIELAB	~65000	Très bonne
CIE94	~97000	Excellente

La norme CIE94 a été choisie comme métrique principale car :

C’est celle qui a la meilleure fidélité à la perception humaine, elle permet aussi une distinction optimale des nuances de couleurs et c’est celle qui avait le meilleur résultat d'étiquetage des biomes.

Pour des résultats rapides, la métrique euclidienne reste une alternative viable.





Validation et interprétation avec l'Indice de Davies-Bouldin

L'indice de Davies-Bouldin mesure la qualité du clustering. Plus elle est proche de 0, plus le résulat est bon. Nous avons implémenté cet algorithme pour K-Means.

Tests effectués avec un nombre de clusters = 6 :

Planète 1 (1400x1400) Euclidien : Score = 0.8428 (bon clustering)

Planète 1 (1400x1400) CIE 94 : Score = 0.7328 (bon clustering)

Planète 2 (2600x1600) CIE 94 : Score = 0.6326 (très bon clustering)

Ces valeurs < 1.0 confirment la pertinence de K-Means pour cette tâche.

3. Détection des Écosystèmes

Choix de l'Algorithme : DBSCAN

Pour les écosystèmes, nous avons choisi **DBSCAN** car contrairement à la partie détection de biomes, on ne peut pas deviner facilement combien d'écosystèmes compose un biome. De plus il peut détecter des clusters avec des formes complexe et écosystèmes peuvent avoir des formes complexes (îles, péninsules, etc…). En plus il est parfait pour ma détection basée sur la densité afin d'identifier des régions qui sont spatialement distinctes

Ici la métrique n'est plus la couleur des pixels mais plutôt leurs positions, donc pour calculer leur distance c'est simple, on utilise la formule euclidienne basique.

Problème de Performance et Solution

Une fois codé, DBSCAN s'est révélé extrêmement lent sur nos images (vu que y'a beaucoup de pixel). L'analyse des écosystèmes de tous les biomes prenait environ 2 heures avec l'algorithme de base, même avec une parallélisation du calcul où chaque biome était traité sur un thread séparé. Cette durée s'explique par la complexité $O(n^2)$ de l'algorithme qui doit comparer chaque pixel avec tous les autres pour trouver ses voisins.

Nous avons donc développé une version optimisée (grâce à votre suggestion pendant l'oral) utilisant une grille spatiale. Cette approche divise l'espace en cellules et limite la recherche de voisins aux cellules adjacentes, réduisant la complexité à $O(n \log n)$ en pratique. Ça devient extrêmement plus rapide : les temps de calcul sont divisés de 5 à 10, rendant enfin l'algorithme utilisable sur des images de grande résolution.

Validation avec le Score de Silhouette

Nous avons implémenté le Score de Silhouette (non vu en cours) pour valider DBSCAN étant donné que Davies-Bouldin est pas du tout adapté à DBSCAN.

- Mesure la cohésion intra-cluster vs séparation inter-clusters
- Valeurs entre -1 et 1 (plus proche de 1 = meilleur)

Résultats observés :

- Scores variant de 0.0587 à 0.6416 selon les biomes
- Les biomes fragmentés (Prairie, Désert) montrent de meilleurs scores
- Les biomes uniformes (Eau profonde) ont des scores plus faibles mais restent cohérents

4. Réutilisabilité des algos

Refactoring

Après la présentation orale, nous avons restructuré le projet pour pouvoir utiliser les algorithmes implémenter pour tous les calculs

1. **Classe PixelData** : Encapsule position (x,y) et la couleur RGB
2. **MetriqueDistance** : Interface pour toutes les métriques (couleur ou position

DBSCAN Optimisé

La version finale détecte le type de données :

- **Positions** : Grille 2D avec cellules adaptatives
- **Couleurs RGB** : Grille 3D avec taille de cellule selon epsilon

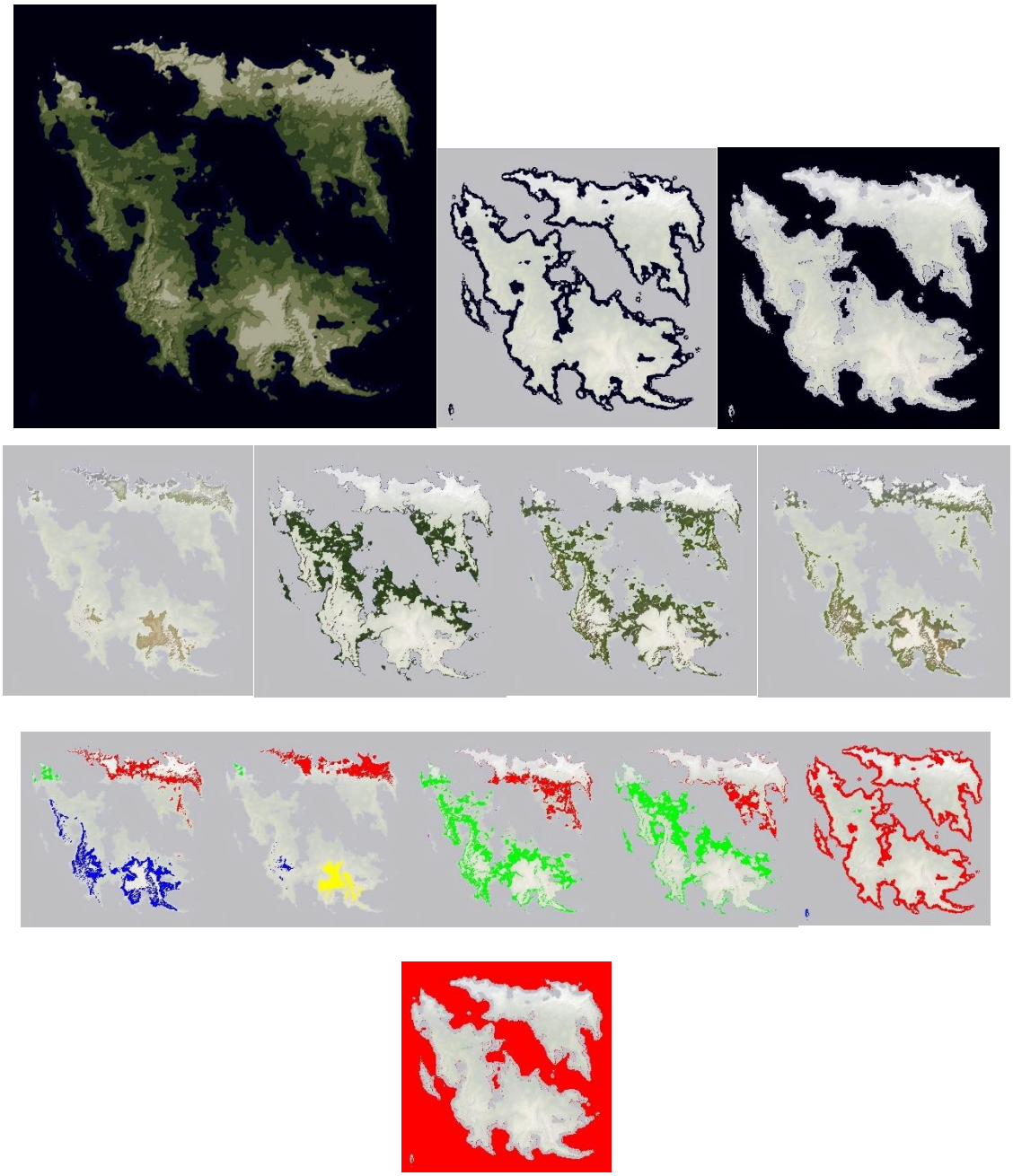
Cette approche permet d'utiliser le même algorithme pour biomes et écosystèmes.

5. Analyse des Résultats

Planète 1 (1400×1400) avec 6 biomes

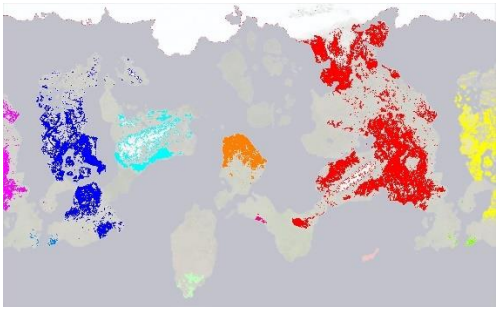
- **6 biomes détectés** : Distribution équilibrée avec "Eau profonde" dominant (54.66%)
- **Écosystèmes** : 3 à 5 par biome, fragmentation modérée
- **Performance** : 78s pour biomes, 168s total pour écosystèmes

Exemple de résultat :



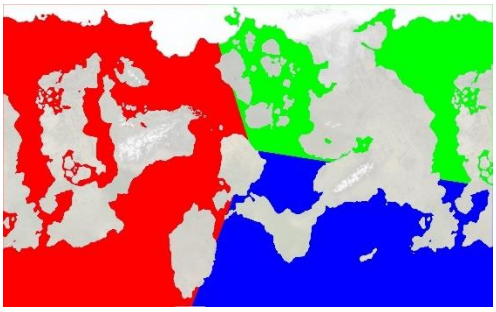
Planète 2 (2600×1600) avec 6 biomes

- **6 biomes détectés** : "Eau profonde" majoritaire (57.18%)
- **Écosystèmes avec DBSCAN** : 2 à 14 par biome, forte variabilité



Exemple de DBSCAN pour écosystèmes Prairies

- **Écosystèmes avec K-Means** : Dépend du nombre de cluster qu'on a demandé (ici 3), plus rapide (1.6s total) mais pas pertinent



Exemple de K-MENS pour écosystème Eau Profonde

Observations Clés

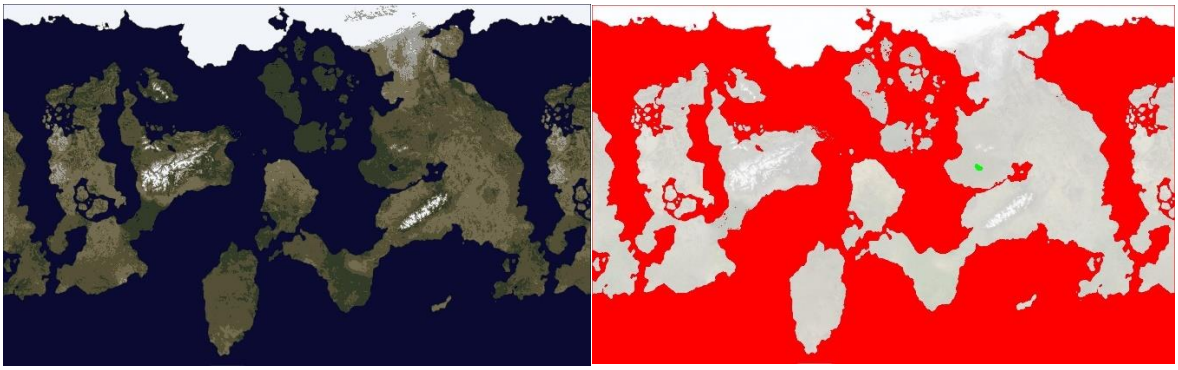
1. **DBSCAN pour biomes** : Inefficace (1 seul cluster détecté ou fragmentation excessive) avec un test, le résultat était une image illisible. Dans tous les cas il est trop long pour cette tache même en version optimisé.
2. **K-Means pour écosystèmes** : Peut être viable mais moins précis que DBSCAN. Il est parfait cependant pour la détection de biomes
3. **Temps de calcul** : Critiques pour DBSCAN non optimisé sur grandes images, acceptable pour DBSCAN Optimisé et rapide pour K-MENS
4. **Indices de validation** : Cohérents avec la qualité visuelle des résultats

6. « IA » pour les Rapports

On a implémenté un système de génération automatique de rapports qui :

- Calcule et interprète les indices de validation
- Suggère la qualité du clustering ("bon", "faible", "mal défini")
- Analyse la fragmentation et la compacité des régions

Exemple de rapport généré :



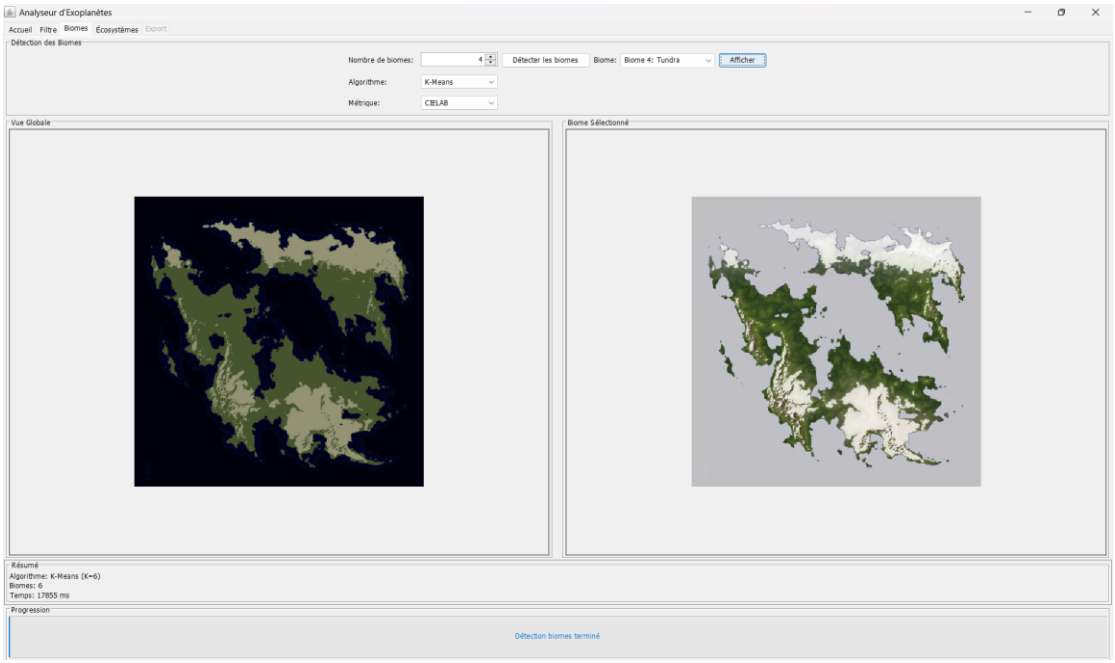
```
=== RAPPORT DES ÉCOSYSTÈMES ===

Biome analysé: Eau profonde
Algorithme utilisé: DBSCAN Optimisé (eps=50.0, minPts=30)
Métrique de distance: Distance Euclidienne - Position
Temps d'exécution: 342429 ms
Nombre d'écosystèmes détectés: 2

=== INDICES DE VALIDATION ===
Score de Silhouette: 0,0000
  → Valeur entre -1 et 1, plus proche de 1 = meilleur
  → Interprétation: Écosystèmes mal définis ou chevauchants

Points de bruit: 10 (0,00%)
  → Un faible pourcentage de bruit (<5%) indique de bons paramètres

=== DÉTAIL DES ÉCOSYSTÈMES ===

Écosystème 0:
  - Nombre de pixels: 2409942 (99,97% du biome)
  - Position centrale: (1268, 898)
  - Étendue spatiale: 2600 x 1600 pixels
  - Compacité: 0,259 (forme étalée ou fragmentée)

Écosystème 1:
  - Nombre de pixels: 757 (0,03% du biome)
  - Position centrale: (1724, 724)
  - Étendue spatiale: 37 x 31 pixels
  - Compacité: 0,345 (forme étalée ou fragmentée)

=== RÉSUMÉ ===
→ Biome peu fragmenté, écosystèmes bien définis
```

Conclusion

Ce projet démontre l'importance du choix approprié des algorithmes selon le contexte :

- **K-Means** : Optimal pour la détection de biomes
    - **Rapide** et adapté, car le nombre de biomes on donne les biomes que op veux trouver (donc le nombre de cluster qu'on cherche)
    - **Euclidienne** pour rapidité, **CIE 94** pour la précision
- **DBSCAN (Optimisé)** : Idéal pour les écosystèmes malgré sa complexité
    - Pas besoin de nombre de clusters défini.
    - Gère le bruit et détecte les formes irrégulières.
    -

Les défis de performance sur grandes images ont été surmontés par des optimisations algorithmiques (grille spatiale). Le projet offre un bon équilibre entre précision et performance, et une bonne analyse des résultats automatisé par des algorithmes de validation.



Interface Graphique (MainIterface)