Mosaïques d’images avec critères avancés

GONZALEZ OROPEZA Gilles
GOUSSEM Ayoub
DUPUIS Thibaut

Plan

1. Présentation du projet
2. Méthodes existantes et implémentées
    a. Etat de l’art
    b. Méthodes implémentées
        i. Statistiques moyennes
        ii. Alignement
        iii. K-Means
    c. Notre implémentation
3. Résultats et démonstration
    a. Analyse
    b. Démonstration
4. Conclusion

Contexte

La transformation d'une image en mosaïque consiste à découper l'image originale en petits blocs (ou tuiles), puis à remplacer chacun de ces blocs par une image différente choisie à partir d'une base d'images, de telle sorte que l'ensemble reconstitue la vue de l'image de départ.

Etat de l'art

- **Méthodes et techniques utilisées :**
  - Extraction de caractéristiques :
    - Statistiques (variance, moyenne, asymétrie, kurtosis, énergie) : [30][31][29]
    - Caractéristiques de Tamura (rugosité, contraste, directionnalité) : [23]
    - Taux de contours (filtre de Canny) : [24]
    - Score d'alignement : [8]
  - Approches intelligentes :
    - K-means clustering (distance Manhattan) : [39][40][41][42]
    - Réseau de neurones à rétropropagation (BPNN) : [43][44]
    - Logique floue hybride avec réseau d'Elman : [45][48][50]
  - Comparaison des approches :
    - Performance en temps de traitement et taux de corrélation.

Sources: "Generating Mosaic Images Based on Texture Analysis", Alaa Yaseen Taqa, 2018.

"Generating photomosaics: an empirical study", Nicholas Tran, 1999.

Méthode des statistiques moyennes

Paramètre principal :

- **Statistique moyenne -> on calcule les moyennes des pixels entre les blocs et les images**

Paramètres supplémentaires :

- **Variance : calcule l'écart type entre les pixels (c'est le meilleur paramètre à utiliser avec la moyenne)**
- **Asymétrie : calculée par**

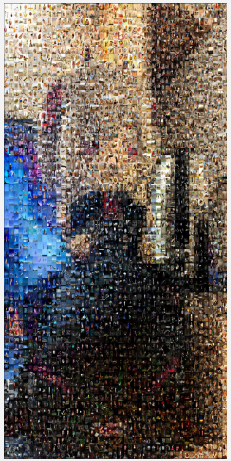
  $s=(MCube[2]-3*MCarré[2]^2+2*M[2]^3, MCube[1]-3*MCarré[1]^2+2*M[1]^3, MCube[0]-3*MCarré[0]^2+2*M[0]^3)$
- **Energie : calcule la somme des (valeurs de niveaux de gris de l'histogramme de l'image en gris)^2**

Paramètre pouvant être associé à toutes les méthodes :

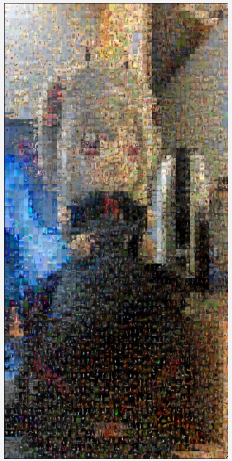
- **Blocs uniques : chaque image ne peut être utilisée qu'une seule fois et elle est placée pour le meilleur bloc possible**

Méthode des statistiques moyennes

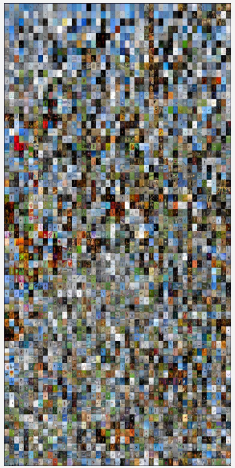
Image originale



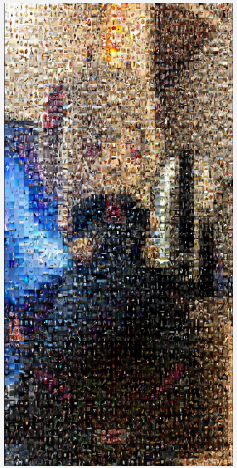
Caractéristique utilisée : Couleur moyenne
PSNR = 12.76 dB



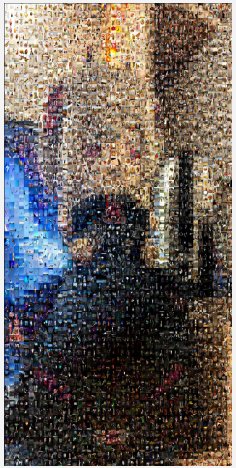
Caractéristiques utilisées : Couleur moyenne et variance
PSNR = 15.77 dB



Caractéristique utilisée : Variance
PSNR = 8.8 dB

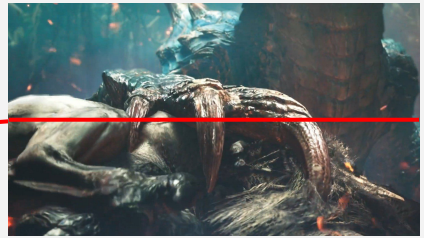


Caractéristique utilisées : Couleur moyenne, <u>asymétrie</u>, <u>énergie</u>
PSNR = 12.76 dB



Caractéristiques utilisées : <u>Asymétrie</u>, <u>énergie</u>
PSNR = 12.76 dB

Méthode de l’alignement



Ligne de **A** (taille : **n**)

	0	1	2	…	n
0	0	-30	-60	…	-30n
1	-30		⋮		⋮
2	-60	…			
⋮			D(i,j)		
n	-30n	…			D(n,n)

Gap = -30

D(i,0) = D(0,i) = i x gap

(Mis)match = - DistanceManhattan(A(i-1), B(j-1))

D(i,j) = max(D(i-1,j) , D(i,j-1) , (mis)match)

Ligne de **B** (taille : **n**)

Score d’alignement des lignes

Alignement(images) = ∑ alignement(lignes)

Méthode des K-Means

K-means -> Créer des clusters sur les pixels des blocs et tester pour chacun des pixels pour quels clusters il est le plus proche
Répéter l'opération jusqu'à l'arrêt.

On choisit alors la meilleure image par rapport au cluster trouvé grâce à la distance euclidienne et on la remplace dans le bloc

Image originale

Image K-Means

Notre implémentation

Choix techniques et données :

- Utilisation de la bibliothèque OpenCV.
- Sélection et importation d'un premier dataset de 120 000 images.
- Importation d'un dataset alternatif (Minecraft) et ouverture à tout dataset de l'utilisateur.
- Adaptation à toute taille d'image et taille de blocs choisie en réduisant la taille de l'image.
- Interface graphique avec FLTK.

Optimisation des calculs :

- Stockages des caractéristiques calculées pour le dataset dans un fichier
- Multi-threading sur certaines méthodes

Résultats et analyse

Image originale
723 x 453 pixels

Blocs de taille 16

Temps d'exécution

Stat moyenne

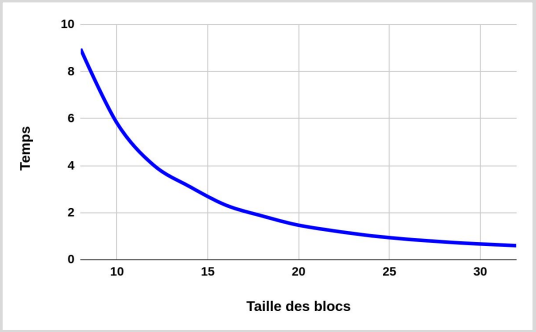
PSNR : 13.8 dB
Temps : 2.3 s

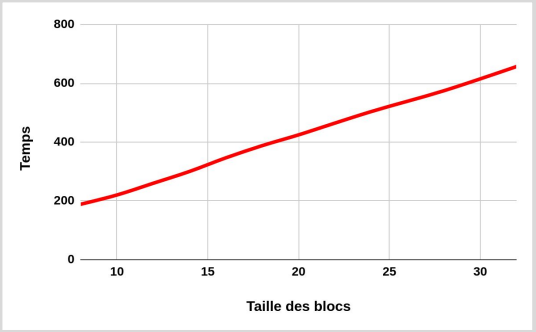
Alignement

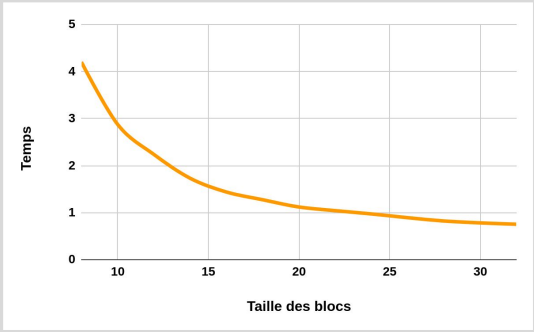
PSNR : 14.5 dB
Temps : 346 s

KMeans

PSNR : 13.2 dB
Temps : 1.4 s



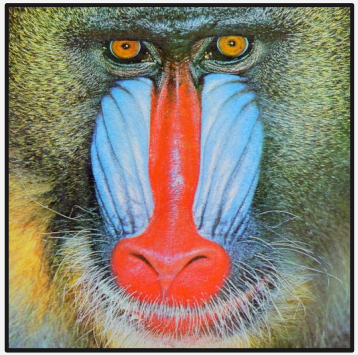




Résultats et analyse

Cas spécial d'alignement plus rapide

Originale

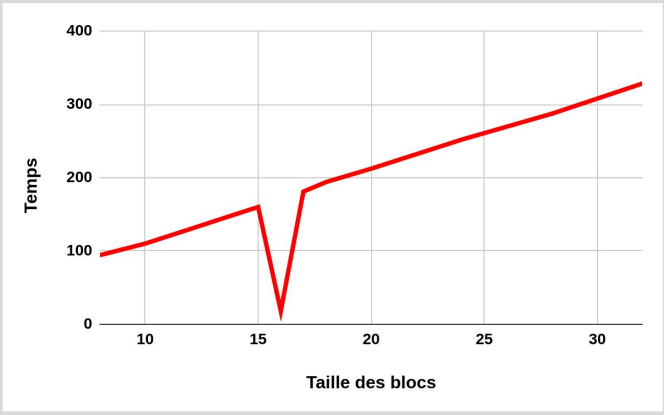


512 x 512 pixels

Alignement

Taille des blocs = 16

PSNR : 14.5 dB
Temps : 18 s

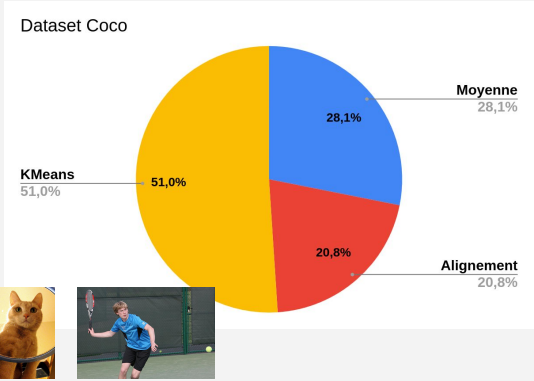


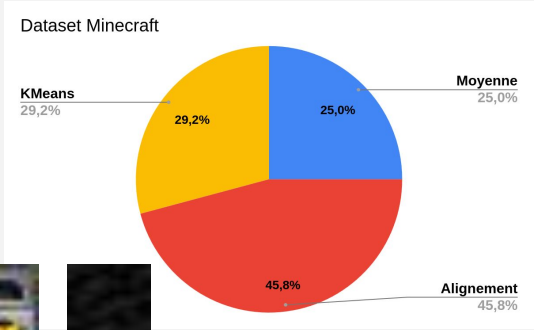
Vrai sur ce PC

-> Tableaux d'alignement rentrent parfaitement dans le cache de ce CPU ?

Résultats et analyse

Sondage, 60 participants

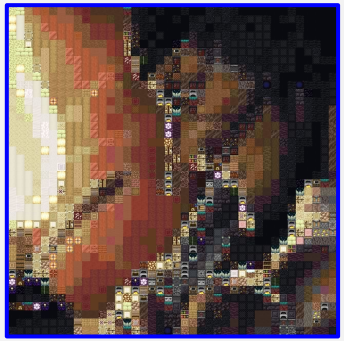


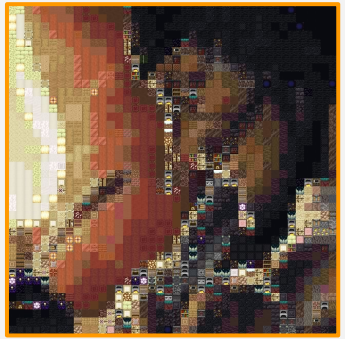


Stat moyenne

Alignement

KMeans





Démonstration

Conclusion

Méthodes avec avantages et inconvénients

Résultats satisfaisants

Optimisation de la méthode par alignement possible

Merci pour votre attention !

Ressources

- Bases de données d'images :
  - *Microsoft COCO Subset*, Roboflow https://public.roboflow.com/object-detection/microsoft-coco-subset/2
  - Pack de textures de Minecraft https://github.com/ZtechNetwork/MCBVanillaResourcePack

- Nicholas Tran (1999). *Generating photomosaics: an empirical study*.
  In *Proceedings of the 1999 ACM symposium on Applied Computing (SAC '99)*, ACM, New York, NY, USA, pp. 105–109.
  https://doi.org/10.1145/298151.298213

- Alaa Yaseen Taqa (2018). *Generating Mosaic Images Based On Texture Analysis*.
  Journal of Education and Science, Vol. 27, pp. 69–101.
  https://www.researchgate.net/publication/336704224_Generating_Mosaic_Images_Based_On_Texture_Analysis