

Objectif : Quantification des couleurs d'une image

Le but de ce devoir est de déterminer automatiquement une palette de couleurs optimale pour une image donnée. Cette palette sera 1) de taille réduite par rapport au nombre initial de couleurs, et 2) la plus représentative possible des couleurs initiales. En effet une image affichée sur un ordinateur peut être encodée sur 8 bits par composantes rouge, verte et bleue (soit 256 valeurs possibles par composante) ainsi potentiellement utiliser $256 \times 256 \times 256 = 16777216$ couleurs. En réalité, beaucoup moins sont utilisées et surtout perceptibles par l'humain. Réduire le nombre de couleur ou réaliser une "*quantification de couleurs*" est une tâche fréquente et c'est une fonctionnalité classique des outils éditeurs d'images (Photoshop, Gimp, etc.) implémentée aussi dans le module Pillow de Python. A noter que cette réduction s'effectue avec perte de couleurs et doit être réalisée avec les bons paramètres (nombre et choix des couleurs) ce qui est votre objectif. La figure ci-dessous illustre le problème à résoudre : étant donnée une image en entrée, proposer une liste de couleurs (que l'on appellera la palette), afin de re-colorier une image en sortie.

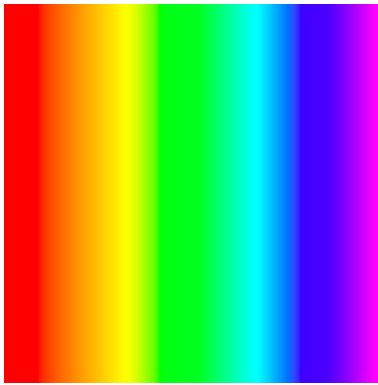


Image originale



Palette de 8 couleurs représentatives

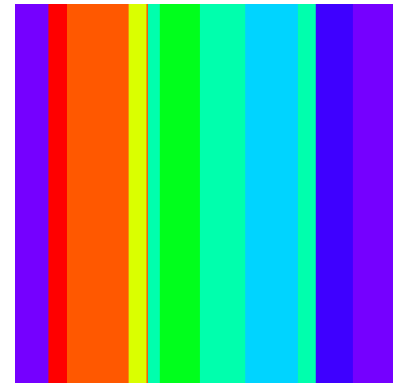


Image originale recoloriée avec la palette

Vous pouvez réaliser ce projet de la manière de votre choix. Vous devez cependant mettre en oeuvre les concepts des TD 2 (utilisation de piles/files/lists), TD 3 (création et parcours d'arbre), et TD 4 (manipulation et dessin d'images). Vous ne pouvez pas utiliser de fonctionnalités permettant de le résoudre avec un outil de traitement d'image ou la fonction `quantize` de Pillow¹. Vous pouvez cependant utiliser des fonctions de chargement/affichage/sauvegarde d'images avec Pillow. D'autres modules externes types Numpy/Pandas ne sont pas autorisés. Vous devez produire un rapport, fournir un code fonctionnel, commenté et justifié décrit en annexe.

Étapes de travail

Voici des étapes de travail suggérées :

1. Prendre en main une image en particulier en la chargeant avec PIL. Lister les couleurs présentes, identifier celles qui sont uniques et leur fréquence.
2. Proposer une méthode (naïve pour commencer) de choix d'une palette de k couleurs. Affichez là sous forme d'image (exemple de d'image au milieu de la figure du dessus) avec une nouvelle image PIL² de taille proportionnelle à la palette.
3. Re-colorier une image avec une palette de k couleurs, et afficher le résultat sous forme d'image PIL. Pour re-colorier chaque pixel, prendre la couleur la plus proche dans la palette en utilisant une fonction de distance (Euclidienne par exemple).

¹Par exemple pour une quantification à 256 couleurs `im = im.quantize(colors=256)`

²Pour créer une nouvelle image avec Pillow : `im2 = Image.new('RGB', (400, 300))`

4. Proposer une méthode de validation de votre approche. Par exemple afficher la différence entre l'image originale et celle re-coloriée. Calculer un score global d'erreur.
5. Améliorer le choix des k couleurs afin de minimiser l'erreur entre l'image originale et re-coloriée. Une piste possible est de trier les couleurs dans une liste, diviser cette liste en k intervals de couleurs et prendre la couleur du milieu de chaque interval. D'autres méthodes plus avancées peuvent être explorées !
6. Tester sur plusieurs images de votre choix ou générées automatiquement avec un nombre et une distribution connue de couleurs. Comparer les performances de vos techniques avec d'autres méthodes (cette fois vous pouvez utiliser un éditeur de texte ou la fonction `quantize` de PIL).
7. Proposer une méthode d'amélioration de calcul de la distance entre deux couleurs, vous pouvez vous baser sur d'autres espaces de couleur³. Cette partie est difficile, les espaces de couleurs possibles sont complexes à comprendre.
8. Optimiser les étapes précédentes (complexité, espace nécessaire, structures de données, etc.).

Annexe : Modalités de rendu du TD

Vous serez évalué sur le rendu de ce TD qui sera à déposer sur Moodle **deux (2) semaines** après les séances d'autonomie et de TD. Vous devrez créer une archive (zip, rar, etc.) nommée `nom1-nom2-inf-tc1-td5.zip` qui contiendra tous les éléments de votre rendu (rapport, code, images de test). Vous pouvez rendre ce rapport seul ou en binôme. Le rendu du TD doit contenir a minima :

1. Toutes les étapes jusqu'à la 6ème doivent avoir été abordées
2. Justifications, illustrations et tests sur plusieurs images

A garder en tête :

- Un code fonctionnel et les tests appropriés devront être fourni dans l'archive qui doit être autonome (le correcteur ne doit pas avoir à rajouter d'image ou de fichier supplémentaire)
- Vous fournirez les images de test et leurs résultat; éviter de prendre cependant des tailles d'images trop importantes.
- Le rapport (maximum 10 pages) comprend :
 - Le détail des étapes que vous avez suivi
 - La description de parties de code difficiles
 - Tout soucis ou point bloquant dans votre code
 - Les graphiques et diagrammes nécessaires
 - Des analyses et discussions en lien avec votre approche
 - Des exemples simples mais aussi difficiles

Tout travail supplémentaire (méthode originale, optimisation poussée) fera l'objet de points en bonus.

³https://fr.wikipedia.org/wiki/Espace_de_couleur