TP CBIR

Nicolas Sidère – 2025

Outils

Pour ce TP, vous devez utiliser Python et la bibliothèque OpenCV

La recherche d'images par le contenu

La recherche d'images par le contenu est un domaine de recherche très actif depuis plusieurs années maintenant. Avec l'explosion de l'imagerie numérique, nous nous retrouvons souvent avec d'énormes bases d'images et la recherche d'une image particulière est un problème difficile à résoudre. Aujourd'hui, la solution la plus souvent retenue est d'annoter manuellement les images avec des mots-clés, ce qui représente un travail long et difficile.

Dans ce TP, vous allez implémenter un petit système de recherche d'images par le contenu.. Le système que vous développerez exploitera plusieurs caractéristiques : d'histogrammes couleurs, formes (Moments de Hu)…

Le système calculera la distance entre plusieurs images d'une base et affichera la liste des images triées de la plus proche à la plus lointaine.

Aucune interface graphique n'est attendue pour ce TP et les résultats seront affichés sous la forme du nom de fichier le plus proche.

En entrée, votre programme devra prendre comme Argument (input) le nom d'une image requête. Alors, le système devra calculer une signature et une distance entre cette image et toutes les images de la base. La sortie de votre programme affichera les N images les plus proches (plus petites distances).

Exemple :

> ***search img035.tif***

Exemple de sortie (avec N=5) :

Image	***Distance***
img078.tif	***0.001***
img032.tif	***0.005***
img005.tif	***0.010***
img098.tif	***0.012***
img062.tif	***0.016***

Les étapes de votre système sont :

- Lire l'image d'entrée
- Calculer les caractéristiques : couleur + Formes
- Supprimer l'image requête de la mémoire (ne garder que sa description)

- Pour chaque image de la base : Répéter les étapes ci-dessus
- Calculer une distance entre le vecteur de l'image requête et chacune des images de la base
- Afficher les N noms de fichiers correspondant aux plus petites distances (images les plus proches)

Pour éviter de surcharger la mémoire, il ne faut jamais avoir plus qu'une image en mémoire à la fois. Seul un tableau contenant les distances et les noms des fichiers est conservé en mémoire. On peut aussi travailler en 2 étapes où (1) toutes les caractéristiques sont pré-calculées et écrites dans un fichier et (2) les distances sont calculées à partir du fichier de caractéristiques.

Description de la base d'images

Vous allez utiliser la base COIL (Columbia University Image Library).
Elle contient 100 différent objets avec 20 images pour chaque objet sous différentes prises de vues.
Elle est librement téléchargeable ici :
http://www.cs.columbia.edu/CAVE/software/softlib/coil-100.php

Caractéristiques

Intersection d'histogrammes couleurs

La première caractéristique que nous allons utiliser est la couleur. Plusieurs méthodes existent pour comparer les couleurs, et nous retiendrons une méthode simple mais qui fonctionne bien, l'intersection d'histogrammes couleurs. Deux étapes sont nécessaires pour utiliser cette méthode

Calcul d'un histogramme

Pour simplifier les calculs, nous utiliserons l'espace RVB pour ce TP, mais sachez que d'autres espaces couleurs peuvent donner de meilleurs résultats de comparaison entre images.
Pour calculer un histogramme couleur, on calcule trois histogrammes, un sur chacun des canaux couleur. On obtient donc 3 histogrammes x 256 valeurs chacun. C'est beaucoup trop pour la comparaison des couleurs, donc nous réduirons cet histogramme à 3 couleurs x *M* valeurs chacun (3 x *M* valeurs au total). La réduction se fait simplement en effectuant une quantification plus forte de l'histogramme. Les valeurs de M utilisées sont couramment de 16, 24 ou 32. Dans ce TP, choisissez la valeur 32.
Pour réduire un histogramme de 256 à M valeurs, on découpe l'histogramme en morceaux de 256/M valeurs, et on additionne les valeurs pour chaque morceau pour obtenir l'histogramme réduit (equal width discretization).

Vous trouverez ici un lien vers la doc OpenCV :
https://docs.opencv.org/master/d8/dbc/tutorial_histogram_calculation.html

Intersection de 2 histogrammes

Pour chaque image, il y a un histogramme réduit de taille 3 x *M* qui est calculé. L'étape suivante est de faire l'intersection (distance) entre 2 histogrammes réduits pour deux images.

Pour cela, vous devez appliquer une comparaison d'histogramme en utilisant la méthode du Chi-2 (ou Chi-square)

Vous trouverez la documentation ici :
https://docs.opencv.org/master/d8/dc8/tutorial_histogram_comparison.html

Calcul des moment de Hu (descripteur de forme)

La deuxième caractéristique que vous allez utiliser pour comparer 2 images est un descripteur de forme : les moments de Hu. Pour chaque image, ces moments vont résumer la forme de l'objet. Ils sont calculés à partir des images en Niveaux de gris

Conversion de l'image en niveau de gris

Vous trouvez ici un tutoriel pour modifier l'espace de couleur :
https://docs.opencv.org/master/df/d9d/tutorial_py_colorspaces.html

Calcul des moments et des caractéristiques

Vous devez ensuite calculer les moments de l'image pour obtenir les moments

Le tutoriel sur le calcul des moments est ici :
https://docs.opencv.org/master/d0/d49/tutorial_moments.html

À partir de ces moments, vous pouvez calculer les 7 caractéristiques de Hu qui formeront votre vecteur de caractéristiques pour comparer les images.

Vous pourrez utiliser la fonction HuMoments de OpenCV :
https://docs.opencv.org/master/d3/dc0/group__imgproc__shape.html#gab001db45c1f1af6cbdbe64df04c4e944

Distance entre les moments

Pour obtenir une distance entre 2 vecteurs de moment, vous pouvez utiliser une distance euclidienne classique

Fonction globale de similarité entre 2 images

Nous avons maintenant 2 distances : couleur et forme. La fonction globale de similarité entre deux images sera la somme pondérée de ces deux distances :

$$Distance = \omega 1 * dist\ couleur + \omega 2 * dist(forme)$$

Avec $\omega 1+\omega 2 = 1$.(par défaut $\omega 1 = \omega 2 = 0{,}5$)