

TD 3 – RECHERCHE D'IMAGES PAR LE CONTENU (CONTENT-BASED IMAGE RETRIEVAL: CBIR)

INTRODUCTION

La recherche d'images par le contenu est une des applications les plus connues et en vogue dans le domaine de la vision par ordinateur. Elle consiste à rechercher à partir d'informations récupérées sur une image requête, d'autres images similaires dans une base de données. La recherche par contenu est opposée à la recherche d'images par concepts (utilisant des mots-clés)

En effet, « par contenu » signifie que la recherche analyse le contenu d'une image au lieu de ses métadatas (mots-clés, tags, description...). Le contenu considéré ici fait référence à la couleur, les formes, les textures ou n'importe quelle autre information qui peut être extraite de l'image elle-même. Pour de nombreuses raisons cette recherche par contenu est préférée à la recherche par concepts (imprécisions ou erreurs d'annotations, temps et ressources nécessaires aux annotations...).

L'intérêt croissant de ce domaine a conduit des géants de la recherche comme Google à développer leur propre systèmes de recherches d'images voir figure 1 et 2.

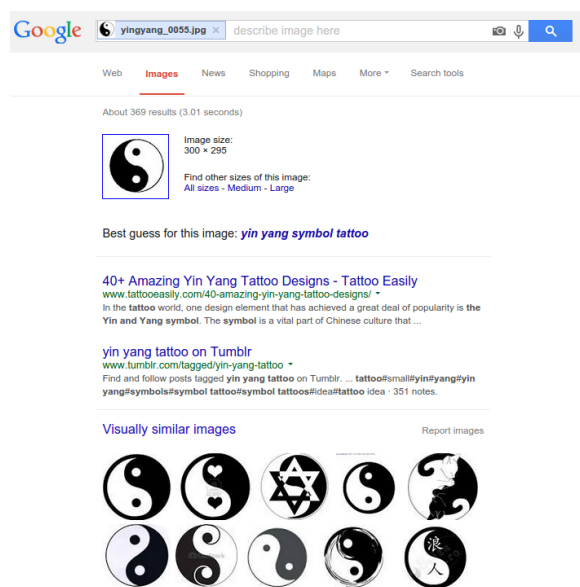


FIGURE 1- RECHERCHE REUSSIE D'UNE IMAGE DE YINYANG SUR GOOGLE IMAGE

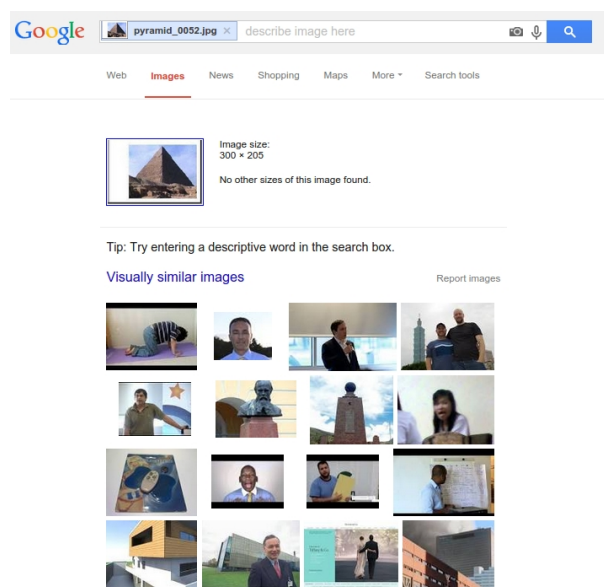


FIGURE 2 - RECHERCHE RATÉE D'UNE IMAGE DE PYRAMIDE SUR GOOGLE IMAGE

C'est ce système de recherche d'images par contenu que vous allez développer dans ce TD.

PRINCIPE GÉNÉRAL

Le principe général de la recherche d'image par le contenu comporte deux étapes. Lors d'une première phase on calcule les « signatures » des images et on les stocke dans une base de données (étape d'indexation). Lors de la seconde phase, l'utilisateur soumet une image comme requête. Le système calcule la signature selon le même mode que lors de

la première phase d'indexation. Ainsi, cette signature est comparée à l'ensemble des signatures préalablement stockées pour en ramener les images les plus semblables à la requête. Voir figure 3.

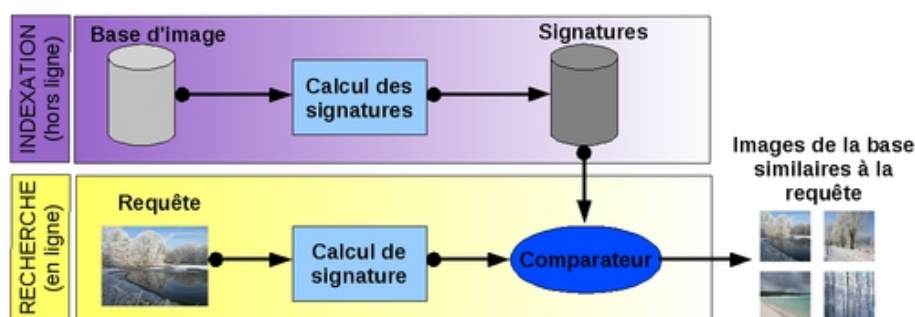


FIGURE 3 - PRINCIPE GÉNÉRAL DE RECHERCHE D'IMAGES PAR CONTENU

Lors de la phase d'indexation, **le calcul de signature** consiste en l'extraction de caractéristiques visuelles des images.

Il est possible d'utiliser différents types de caractéristiques visuelles (couleur, texture, formes), ici nous allons utiliser les **descripteurs SIFT**. Ces descripteurs, créés par David Lowe en 1999, sont très populaires en traitement d'image, car ils sont (pas totalement bien sûr) invariants aux changements de luminosité, de rotation et de taille. De plus ils sont extraits dans des zones d'intérêt d'une image (des zones non banales comme les coins d'un objet, pas un ciel bleu par exemple), voir figure 4.

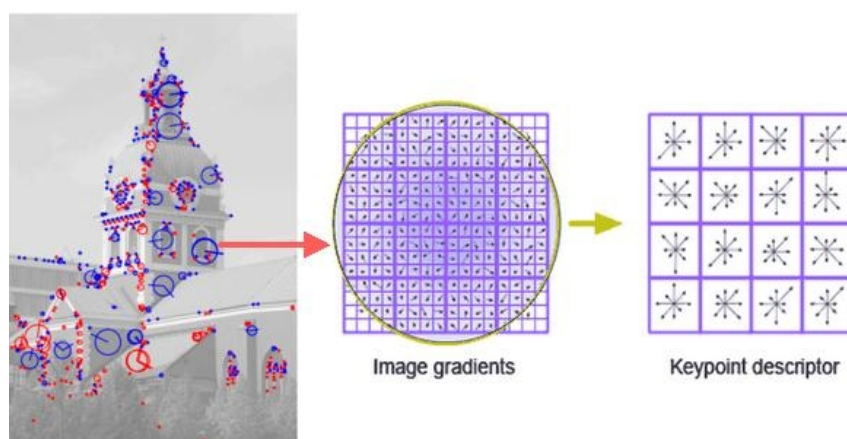


FIGURE 4 - REPRÉSENTATION DE DESCRIPTEURS SIFTS

A l'inverse d'histogrammes de couleur par exemple, l'algorithme SIFT ne donne pas une description globale de l'image. Au contraire les descripteurs détectés sont locaux avec une dimension de 128 chacun. Pour une image, la sortie de l'algorithme SIFT retourne donc une matrice de taille $128 \times N$, où N représente le nombre de descripteurs détectés.

Comme la recherche d'image par contenu a besoin d'un descripteur « global » de l'image pour comparer les données visuelles, il faut transformer l'information contenue dans tous ces descripteurs locaux en un descripteur global de l'image.

Il existe plusieurs méthodes pour faire cela et nous allons utiliser ici la méthode appelée « **Bag of Visual Words (BOVW)** ».

« Bag of Words » fut au départ inventé pour le domaine de la recherche de textes par contenus. En effet un texte était représenté par un histogramme des mots qu'il contenait. Deux textes ayant des histogrammes avec des pics pour les mots « dents » et « opération » avaient plus de chance de correspondre qu'avec un autre texte ayant un histogramme avec des pics pour « voiture » et « clés ».

C'est ce même principe qui est utilisé dans Bag of Visual Words, voir figure 5.

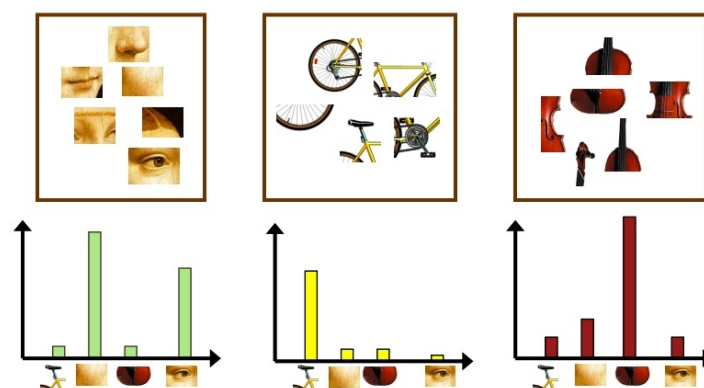


FIGURE 5 - ILLUSTRATION DE LA CRÉATION D'UNE "SIGNATURE" D'UNE IMAGE, C'EST A DIRE UN HISTOGRAMME DE MOTS VISUELS

Bien sûr, des « mots visuels » ça n'existe pas. C'est pour cela que l'on doit créer ce qu'on appelle un **dictionnaire visuel** auparavant. Pour cela il faut prendre une base de données d'images dans laquelle on extrait tous les descripteurs M SIFTS qui seront clusterisés en un nombre $K \ll M$ de mots visuels.

Dans la figure 6, on considère $M=14$ descripteurs à deux dimensions (c'est pour illustrer, ne pas oublier qu'un descripteur SIFT a 128 dimensions) qui, après clusterisation, donne $K=4$ mots visuels.

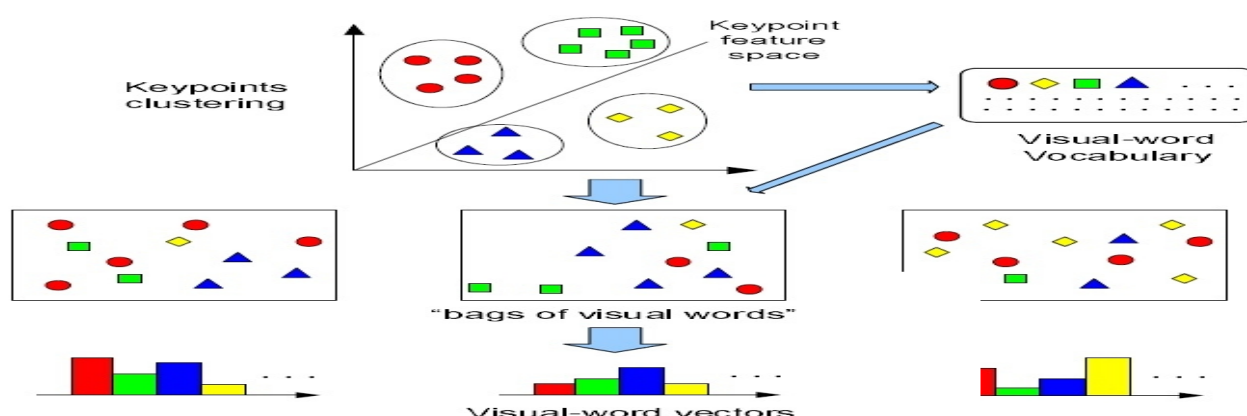


FIGURE 6 - ILLUSTRATION DE LA CREATION DE MOTS VISUELS PAR CLUSTERISATION

1 - CRÉATION DU DICTIONNAIRE DE MOTS VISUELS

Afin de vous permettre d'aller plus vite lors de cette séance. Il vous est demandé de lire le paragraphe en bleu suivant pour comprendre la suite, mais vous ne devez pas

l'implémenter. A la place vous pouvez directement charger les clusters (les dictionnaires visuels) qui se trouvent dans le dossier *centers* (le chargement des clusters est déjà implémenté dans le *main.m*).

La première étape primordiale consiste à calculer ce dictionnaire de mots visuels. Pour cela il faut extraire et stocker tous les descripteurs SIFT **de toutes les images** de la base de données dans une matrice de dimension $128 \times M$. Pour extraire les descripteurs SIFT **d'une seule image** stockée dans une variable appelée *cur_image*, utiliser la fonction fournie *functionKeypointsdetectionprogram* :

```
[K, D] = functionKeypointsdetectionprogram(single(cur_image)) ;
```

Une fois que cette matrice de tous les descripteurs est extraire, la clusterisation se fait en utilisant l'algorithme de **K-means**. Cet algorithme est aussi implémenté dans la bibliothèque fournie **vlfeat**. Il faut lui spécifier le nombre de clusters souhaités K en sortie (ce nombre de clusters a été défini par défaut à 200 dans la variable *nb_cluster*).

```
[centers, assignments] = vl_kmeans(double(all_descriptors), nb_clusters);
```

Cette étape est à effectuer une seule fois pour trouver un dictionnaire de K mots visuels définitifs. Pensez à l'enregistrer dans un fichier (fonction *save*). Lorsque vous continuerez la suite du TD, utiliser la fonction *load* pour recharger ce dictionnaire.

2- CRÉATION DE LA BASE DE DONNÉES DE SIGNATURES.

Maintenant que le dictionnaire de mots visuels est créé, il est possible d'obtenir la signature de chaque image de la base de données et d'enregistrer cet ensemble de signatures. Pour créer la signature d'une image, il faut, comme précisé dans la partie Principe Général, calculer l'histogramme des occurrences des différents K mots visuels. Pour une image la signature est calculée de la manière suivante :

```
Extraire les descripteurs SIFT de l'image
Créer un histogramme vide de  $K$  bins
Pour chaque descripteur SIFT de l'image {
    Vérifier de quel mot visuel il est le plus proche (distance euclidienne)
    Rajouter +1 dans le bin correspondant de l'histogramme
}
Normaliser l'histogramme par le nombre de descripteurs SIFT
```

Cette étape est à effectuer une seule fois pour trouver toutes les signatures de la base de données. Vous pouvez les stocker dans une matrice de taille $K \times nb_{images}$ qu'il est possible d'enregistrer dans un fichier. Lorsque vous continuerez la suite du TD, vous n'aurez comme ça pas à recalculer toutes les signatures.

3-RECHERCHE AVEC UNE IMAGE REQUÊTE

Maintenant que le dictionnaire de mots visuels et l'ensemble des signatures ont été calculées, il est possible d'effectuer une recherche par contenu d'image.

La phase de comparaison consiste généralement à définir une mesure de similarité globale entre deux images. Au moyen de cette mesure de similarité et d'une image requête, on peut alors calculer l'ensemble des mesures de similarités entre cette image requête et l'ensemble des images de la base d'images. On peut alors ordonner les images

de la base suivant leur score, et présenter le résultat à l'utilisateur, les images de plus grand score étant considérées comme les plus similaires.

Commencez par calculer la signature de votre image requête (descripteurs SIFTS -> histogramme).

Comme précisé précédemment il existe plusieurs mesures de similarité. Celle que vous allez implémenter s'appelle l'intersection d'histogrammes. L'intersection calcule un score de ressemblance spécialement adapté aux histogrammes (valeur maximale de 1 si les histogrammes ont bien été normalisés à l'étape 2).

L'équation suivante calcule l'intersection $S(a, b)$ entre un histogramme a et b à K bins.

$$S(a, b) = \sum_{i=1}^K \min(a_i, b_i)$$

Vous pouvez maintenant trier toutes les scores du plus au moins fort (fonction sort dans Matlab).

Afficher les images ayant les plus fort scores pour vérifier si le recherche fonctionne. Vous pouvez changer d'image requête en changeant le numéro d'image à la ligne 19.

4] ETUDES DES PERFORMANCES

Se rendre compte visuellement que votre algorithme marche ne signifie malheureusement que peu. Dans cette partie vous allez implémenter un système de métrique permettant de mesurer les performances. Vous aller ainsi pouvoir faire varier le nombre de clusters (de mots visuels dans votre dictionnaire) pour vous rendre compte de l'évolution de ces performances.

Il est conseillé à ce stade de copier le contenu de votre *main.m* dans un nouveau fichier Matlab sur lequel vous allez pouvoir faire des modifications.

Calcul des performances :

Si l'on considère le nombre de cluster K comme fixe, la performance de votre algorithme se mesure par le calcul de la moyenne de la **précision aux 10 plus proches** pour chaque image requête.

Pour calculer la précision aux 10 plus proches pour une image requête :

- Récupérer le nom de catégorie de l'image requête (« canon », « pyramid », ...). Le nom de catégorie est contenu dans le titre de l'image (pour vous aider, regarder la fonction Matlab strsplit)
- Pour les dix premières images retournées de plus haut score (sauf la première car cela devrait être la même image que l'image requête, donc de l'image n°2 à 11) vérifier le nombre TP d'images ayant la même catégorie que votre image requête.
- La précision de votre image requête $P_{cur} = TP/10$

Faites ceci en prenant toutes les images de la base comme images requête et moyenner pour obtenir la précision moyenne pour le nombre de clusters K .

Travail demandé : Faites varier les valeurs du nombre de mots visuels K entre les valeurs suivantes : [5, 10, 200, 1000, 16000] pour votre dictionnaire et observer les performances. Afficher les courbes de valeurs de performances, qui vous seront utiles pour la partie suivante.

5] RAPPORT

Pour ce dernier TD, il vous est demandé d'écrire un **rapport concis** contenant les parties suivantes :

- Introduction
- Montrer quelques exemples d'images retournées pour différentes images recherchées. Donner une explication de pourquoi il semble normal que certains résultats « faux » aient été retournés.
- Présentation des résultats et **interprétation** (comment évoluent les performances et en donner la raison).
- Conclusion (problèmes rencontrés ou résolus, possibles perspectives,...)

Envoyez votre rapport à souad.chaabouni@u-bordeaux.fr dans une archive contenant aussi **le code Matlab** pour le dimanche 17 décembre 2017.