

UNIVERSITÉ NATIONALE DU VIETNAM À HANOÏ  
INSTITUT FRANCOPHONE INTERNATIONAL

---



Option : Systèmes Intelligents et Multimédia (SIM)

*Promotion : XXII*

INDEXATION MULTIMODALE DES CONTENUS

**PROJET : OPTIMISATION DU TP CBIR**

Jean Claude SERUTI ZAGABE

Azaria ALLY SAIDI

**MASTER II**

Encadrant :

Nicolas SIDERE

Année académique 2018-2019

# Table des matières

<b>0</b>	<b>Introduction</b>	<b>2</b>
0.1	Contexte . . . . .	2
0.2	Objectif . . . . .	2
<b>1</b>	<b>Première partie : évaluation du système</b>	<b>2</b>
1.1	Descripteurs Globaux sur les images . . . . .	3
1.2	Fonctionnement du programme . . . . .	4
1.3	Expérimentation . . . . .	6
<b>2</b>	<b>Deuxième partie : Optimisation de l'indexation</b>	<b>6</b>
2.1	Expérimentation . . . . .	6
<b>3</b>	<b>Conclusion et Perspective</b>	<b>9</b>

# 0 Introduction

## 0.1 Contexte

Dans le cadre de l'évolution technologique, la recherche d'images par le contenu est devenu un domaine de recherche plus fréquent et nous assistons à l'accroissement très rapide de l'imagerie numérique, qui constitue une masse d'information, d'où la recherche d'une image devient la problématique actuelle à résoudre.

A ce jour, les études prouvent que, l'une des solutions la plus efficace est d'annoter manuellement ces images avec des mots clés. Ce processus d'annotation représente un travail fatidique et difficilement accompagnable des approches. Partant de cette hypothèse, notre projet, consiste à l'implémentation d'un système de recherche d'images par le contenu en se focalisant sur l'approche de descripteurs globaux (Texture, formes et couleurs).

L'idée alors est de calculer les caractéristiques pour chaque image en entrée et de rechercher les images caractéristiques les plus semblables. Ainsi dans ce rapport, nous allons présenter le travail détaillé réalisé et les différentes méthodes utilisées, ensuite les résultats obtenus, en deux grandes parties, dont la première partie qui concerne l'évaluation du système et la deuxième partie qui est l'optimisation de l'indexation.

## 0.2 Objectif

Les objectifs poursuivis dans ce travail sont :

- L'évaluation du système en implémentant le calcul de la F-Mesure après avoir obtenu les indices de l'évaluation des performances du système dont le Rappel et la Précision.
- L'Optimisation de l'indexation en intégrant l'algorithme de clustering pour permettre d'accélérer la phase de recherche en utilisant la méthode de création de cluster(k-means).

# 1 Première partie : évaluation du système

Un des problèmes fondamentaux dans le domaine des bases de données multimédias, réside dans la recherche de similarité, c'est-à-dire, le besoin de chercher un petit ensemble d'objets qui soient similaires ou très peu rapproché d'un objet requête donné. La plupart des méthodes de recherche d'images par le contenu sont constituées des phases suivantes :

- Une étape d'indexation d'image : consiste à extraire des signatures compactes de leur contenu visuel. Ces signatures se présentent sous la forme de vecteurs multidimensionnels appelés descripteurs.
- Une étape de structuration de l'espace de description : il s'agit dans cette étape de mettre en place une structure d'index multidimensionnels permettant une recherche efficace des dizaines voire des centaines de milliers d'images.

- Une recherche de similarité : dans la plupart des méthodes, une distance est associée à chaque descripteur et une recherche des k plus proches voisins est effectuée.

Nous nous sommes focalisés dans le cadre de ce projet principalement sur les points 2 et 3 pour pouvoir répondre efficacement aux problématiques liées à la recherche des grandes bases d'images. Les travaux ont consisté, d'une part à améliorer la performance des méthodes d'indexations basées sur l'approximation en tenant compte des différentes descriptions des images afin d'apporter des réponses au problème du passage à l'échelle.

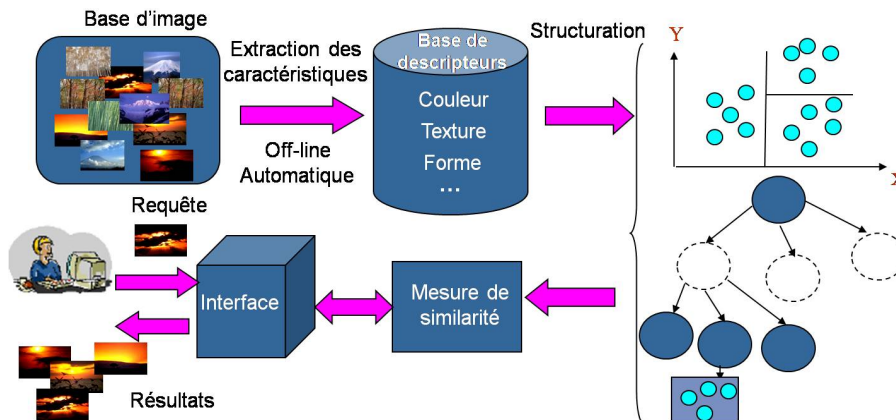


FIGURE 1 – *Système de recherche*

## 1.1 Descripteurs Globaux sur les images

Description globale de l'image est une description approximative de toute l'image, elle considère l'image dans son ensemble, caractérise l'image en utilisant des statistiques calculées sur l'image entière et une description moins fine de l'image notamment de recherche des objets.

En indexation d'images, cette apparence visuelle globale est calculée sur toute l'image peut être résumée par trois caractéristiques constituant ainsi ses descripteurs dont :

- La couleur,
- La texture,
- La forme.

Ces descripteurs sont enregistrés dans des fichiers dont les noms correspondents à chaque image de la base c'est-à-dire pour une image donnée de la base correspond à un fichier du même nom que l'image dans le dossier feature et qui comprends en son sein les trois descripteurs globaux de la dite image. Ces descripteurs ont été enregistré dans les fichiers grâce au module « Pinkle » de python, permettant de sauvegarder dans un fichier, au format binaire, n'importe quel objet Python.

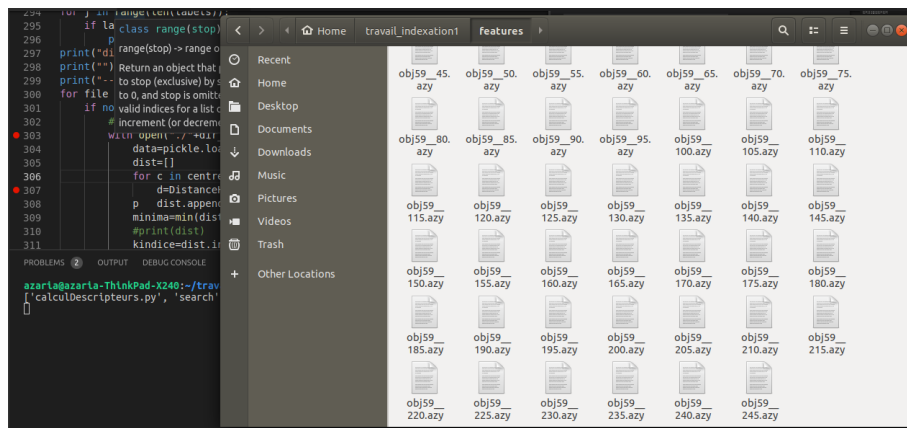


FIGURE 2 – Descripteurs des images

## 1.2 Fonctionnement du programme

Pour lancer notre programme, il faudrait bien se positionner au niveau du terminal ou de l'EDI pour l'exécuter. De notre côté nous expérimentons par l'EDI Visual Studio Code en tapant « `python Descripteurs.py search obj2_50.png` ».

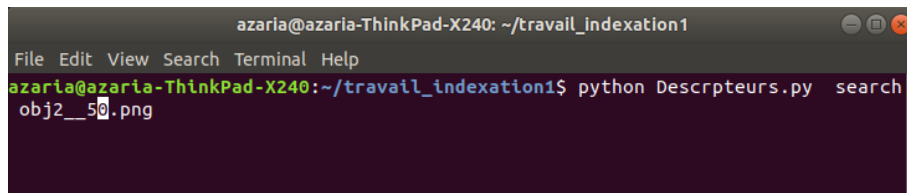


FIGURE 3 – Requête du programme

La figure 3 représente la requête utilisé pour faire la recherche de l'image dans la base de données Coil 100 qui contient 50 objets par classe de 20, soit un total de 1000 images différentes.

A l'issue de ce programme nous obtenons les résultats ci-après que nous essayerons d'interpréter.

```

azaria@azaria-ThinkPad-X240:~/travail_indexation1$ python Descripteurs.py search
obj2__50.png
['Descripteurs.py', 'search', 'obj2__50.png']
('sommaton des distances :', 0.90026760992104449)
('sommaton des distances :', 0.61360497189784491)
('sommaton des distances :', 0.59704305307757621)
('sommaton des distances :', 1.1940031563706013)
('sommaton des distances :', 1.1295024818768669)
('sommaton des distances :', 1.4392040617438506)
('sommaton des distances :', 1.1966806117853086)
('sommaton des distances :', 0.59751880122810297)
('sommaton des distances :', 1.3627200984023666)
('sommaton des distances :', 0.79295673823961743)
('sommaton des distances :', 0.78930597557980919)
('sommaton des distances :', 0.83407145004209404)
('sommaton des distances :', 0.6757718260526353)
('sommaton des distances :', 0.47591995318358238)
('sommaton des distances :', 0.50395999207639242)
('sommaton des distances :', 0.26369565969828107)
('sommaton des distances :', 0.84708836689859812)
('sommaton des distances :', 1.2662201738088976)
('sommaton des distances :', 0.70513378487960821)
('sommaton des distances :', 0.80702231297324389)
('sommaton des distances :', 0.58024419725059373)

```

FIGURE 4 – Affichage des distances calculées

Cette figure 4 affiche les distances calculées en utilisant l'algorithme Knn, qui veut dire k plus proche voisins. Partant de cette méthode, le système détecte et sélectionne les distances les plus proches afin de déduire la classe de l'image en entrée, ci-dessous en voici le résultat obtenu avec 5 plus proches voisins pour 8 bit histogramme choisi.

```

azaria@azaria-ThinkPad-X240: ~/travail_indexation1
File Edit View Search Terminal Help
('sommaton des distances :', 0.59615090226040812)
('sommaton des distances :', 0.96267793083678233)
('sommaton des distances :', 0.93402708730697515)
('sommaton des distances :', 0.4142128927568266)
('sommaton des distances :', 0.66102855003325089)
('sommaton des distances :', 0.3582421222246559)
('obj2__50.azy', '=', 0.0)
('obj2__40.azy', '=', 0.08239359018142664)
('obj2__45.azy', '=', 0.085201979289114049)
('obj2__75.azy', '=', 0.085711628112666985)
('obj2__110.azy', '=', 0.087044895507998432)
-----
-----
-----
('voici les k voisin:', ['obj2__50.azy', 'obj2__40.azy', 'obj2__45.azy', 'obj2__75.azy', 'obj2__110.azy'])
(1.0, ' et ', 0.013888888888888888)
('voici le nombre:', 5, ' et la precision:', 1.0, ' nbre element voisin ', 5)
('le recall:', 0.013888888888888888, ' et la mesure F est: ', 0.0273972602739726)
Calcul de la distance globale(histogramme 8 bits + moment de hu)
azaria@azaria-ThinkPad-X240:~/travail_indexation1$

```

FIGURE 5 – Affichage des distances calculées

On remarque sur la figure 5 que la précision du système est de 100% cela est du au fait que les images renvoyées sont toutes de la même classe que l'image en entrée et la F-Mesure est de 0.0274.

### 1.3 Expérimentation

Avec quelques paramètres différents à tester, nous mesurons leur influence sur la performance de notre système et les conclusions obtenues sont les suivantes :

- **Influence de N : nombre de documents retournés. Valeurs possibles : 5, 20, 50, 100** : suite à ces paramètres, il aura aucune influence, les nombres de documents retournés dépendent de K plus proches voisins.
- **Influence de M : taille de l'histogramme. Valeurs possibles : 16, 64, 128, 256** : suite à ces paramètres les petites valeurs permettent d'obtenir plus rapidement le résultat que les grandes valeurs. Donc, on remarque que les temps d'exécution de notre système devient plus long en utilisant ces valeurs.
- **Influence des poids de pondération des distances  $v$  (pour renforcer l'impact de caractéristiques** : une forte influence par rapport au k voisins dépendant de toutes les caractéristiques.

## 2 Deuxième partie : Optimisation de l'indexation

Dans cette partie, nous allons effectuer une classification des différentes images en les affectant à des centres que nous avons préalablement défini.

A partir des centres défini, On affecte aléatoirement chaque image à l'un des centres et on itère comme suit : les centres des différents groupes sont recalculés et chaque image est de nouveau affecté à un centre en fonction du centre le plus proche c'est-à-dire le centre avec le quel la distance est moins élevée. La convergence est atteinte lorsque les centres sont fixes.

### 2.1 Expérimentation

Afin de pouvoir évaluer notre programme, nous avons effectuer une classification de la base d'image en utilisant le descripteur de forme. Pour des raison de temps de calcul, nous avons fixé le nombre de bits à 8 bits. Nous avons tester avec différentes valeurs de k et les résultats illustrés ci-dessous.

```

262 kmean=KMeans(n_clusters=10,random_state=0)
263 kmean.fit(m)
264 centres=kmean.cluster_centers_
265 labels=kmean.labels_
266 print("les centres:",centres)
267 #print(len(centres))
268 print(labels)
269 #print(fichiers)
270 #print("voici le HU de :",hu)
271 #retour=kmean.fit_predict(hu)
272
273 #print("voici le retour de la fonction predict:",retour)
274 for mom in centres:
275     d=DistanceHu(hu,mom)
276     distances.append(d)

```

FIGURE 6 – *Extrait code Kmeans*

Partant de cette capture de la figure 6 qui représente l'extrait de code ajouter pour créer le cluster en implémentant l'algorithme de k-means.

```

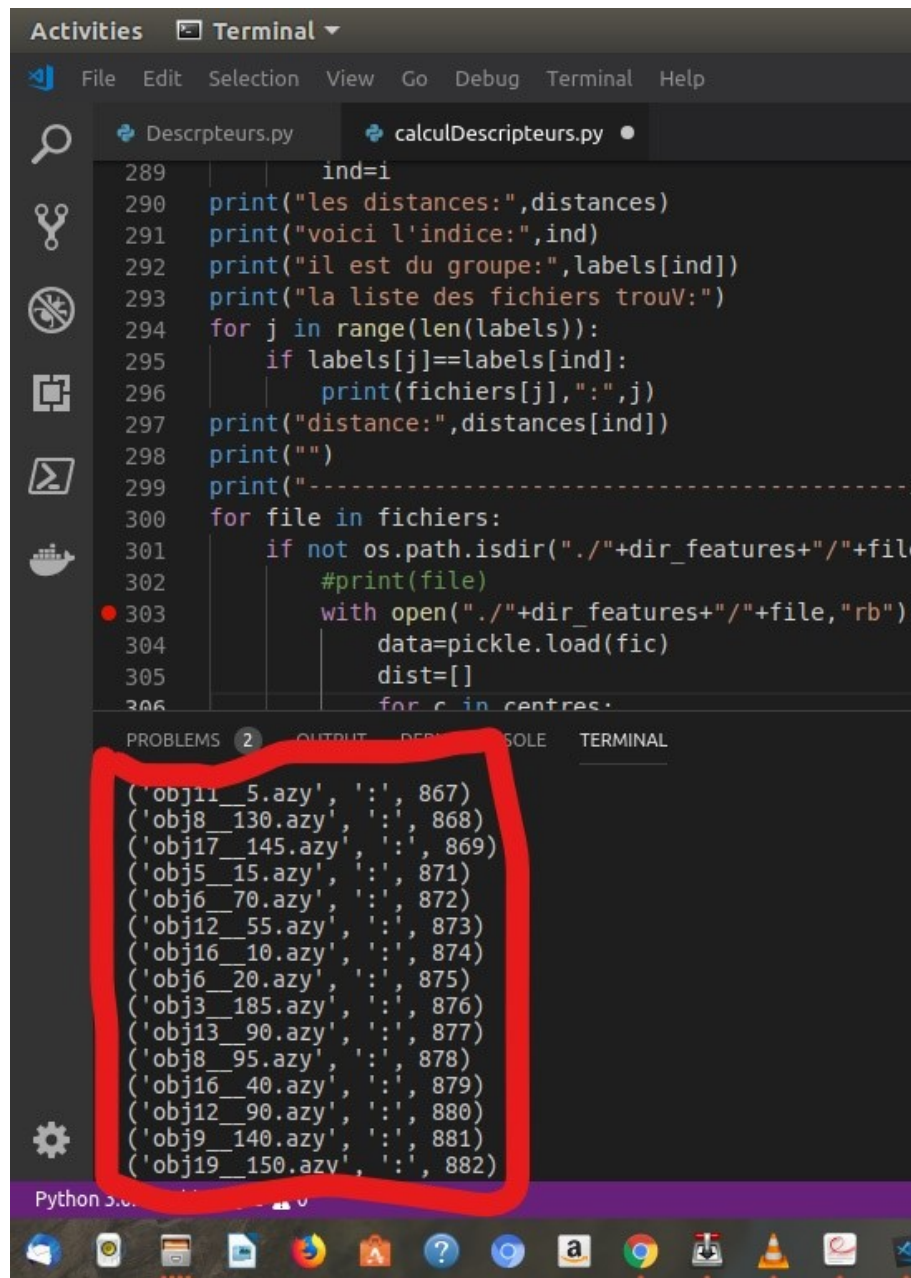
azaria@azaria-ThinkPad-X240:~/travail_indexation$ python calculDescripteurs.py search obj2_50.png
['calculDescripteurs.py', 'search', 'obj2_50.png']
les fichiers existents déjà
array([ 1.95384305e-03,  1.57745464e-07,  2.44034137e-11,
        6.57663944e-11, -7.01002558e-22, -1.98614382e-14,
        2.53973323e-21]], array([ 2.21338011e-03,  3.67021079e-07,  1.03262425e-11,
        3.31940899e-11, -3.69770795e-22,  8.60047308e-15,
        -4.93875488e-22]], array([ 1.08150383e-03,  7.75188738e-09,  2.78406423e-12,
        1.41776996e-11,  7.63324493e-23, -1.18198144e-15,
        4.59068279e-23]], array([ 1.91786475e-03,  1.32405618e-07,  2.16795827e-11,
        1.26963922e-10,  4.23298192e-21,  3.44488632e-14,
        -5.14316222e-21]], array([ 1.87014549e-03,  1.37987596e-07,  1.77966766e-11,
        1.34624236e-10,  5.48995960e-21,  3.21843199e-14,
        -3.64247011e-21]], array([ 2.15139676e-03,  3.91044748e-07,  7.16495591e-12,
        2.26443455e-11, -1.78241821e-22,  7.53709634e-15,
        -2.32834452e-22]], array([ 1.05919488e-03,  7.24579309e-09,  3.69329230e-12,
        1.31912650e-11,  7.81444752e-23, -1.01086590e-15,
        4.86933558e-23]], array([ 1.96650658e-03,  1.53732592e-07,  2.53689010e-11,
        7.54370180e-11, -8.09397702e-22, -2.19654792e-14,
        3.11729599e-21]], array([ 1.04424102e-03,  7.29768881e-09,  2.90987781e-12,
        1.13378962e-11,  5.15228753e-23, -9.05919397e-16,
        3.98300329e-23]], array([ 1.96516302e-03,  1.52937239e-07,  2.48165936e-11,
        7.74867834e-11, -1.21440706e-21, -2.15255842e-14,
        3.11249080e-21]], array([ 1.01196216e-03,  1.39569896e-07,  1.53034024e-11,
        1.35274755e-10,  5.79816516e-21,  2.91367740e-14,
        -2.06488468e-21]], array([ 2.05031334e-03,  4.21444705e-07,  6.37878132e-13,

```

FIGURE 7 – *Cluster- Kmeans*

La capture de la figure 7 représente les différents centres sur le jeu des données descripteurs de formes(Moments de Hu).





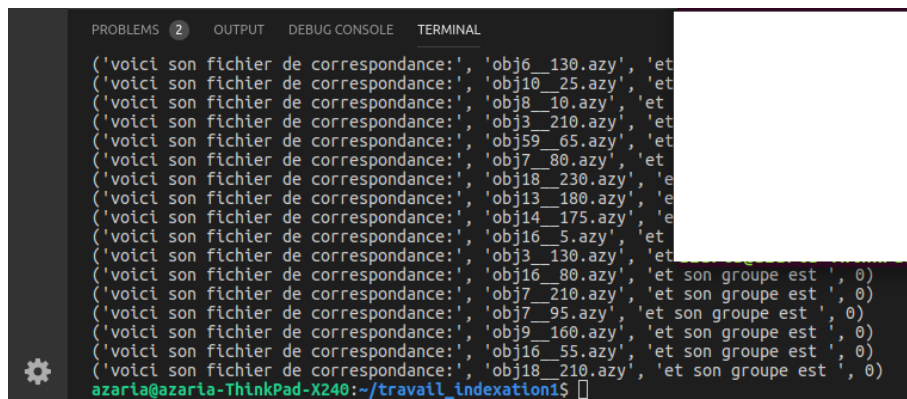
```
289     ind=i
290     print("les distances:",distances)
291     print("voici l'indice:",ind)
292     print("il est du groupe:",labels[ind])
293     print("la liste des fichiers trouv:")
294     for j in range(len(labels)):
295         if labels[j]==labels[ind]:
296             print(fichiers[j],":",j)
297     print("distance:",distances[ind])
298     print("")
299     print("-----")
300     for file in fichiers:
301         if not os.path.isdir("./"+dir_features+"/"+file):
302             #print(file)
303             with open("./"+dir_features+"/"+file,"rb") as fic:
304                 data=pickle.load(fic)
305                 dist=[]
306                 for c in centres:
```

```
('obj11_5.azy', ':', 867)
('obj8_130.azy', ':', 868)
('obj17_145.azy', ':', 869)
('obj5_15.azy', ':', 871)
('obj6_70.azy', ':', 872)
('obj12_55.azy', ':', 873)
('obj16_10.azy', ':', 874)
('obj6_20.azy', ':', 875)
('obj3_185.azy', ':', 876)
('obj13_90.azy', ':', 877)
('obj8_95.azy', ':', 878)
('obj16_40.azy', ':', 879)
('obj12_90.azy', ':', 880)
('obj9_140.azy', ':', 881)
('obj19_150.azy', ':', 882)
```

FIGURE 8 – *Parcours descripteurs*

La figure 8 illustre comment le programme parcourt les objets par groupes pour obtenir les images correspondantes et la plus proches, c'est- à-dire un parcours de descripteurs dans la base des données

La figure 9 illustre la correspondance des différentes objets selon leur groupe à partir de partitionnement de k-means.



```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL
('voici son fichier de correspondance:', 'obj6_130.azy', 'et
('voici son fichier de correspondance:', 'obj10_25.azy', 'et
('voici son fichier de correspondance:', 'obj8_10.azy', 'et
('voici son fichier de correspondance:', 'obj3_210.azy', 'et
('voici son fichier de correspondance:', 'obj59_65.azy', 'et
('voici son fichier de correspondance:', 'obj7_80.azy', 'et
('voici son fichier de correspondance:', 'obj18_230.azy', 'e
('voici son fichier de correspondance:', 'obj13_180.azy', 'e
('voici son fichier de correspondance:', 'obj14_175.azy', 'e
('voici son fichier de correspondance:', 'obj16_5.azy', 'et
('voici son fichier de correspondance:', 'obj3_130.azy', 'et
('voici son fichier de correspondance:', 'obj16_80.azy', 'et son groupe est ', 0)
('voici son fichier de correspondance:', 'obj7_210.azy', 'et son groupe est ', 0)
('voici son fichier de correspondance:', 'obj7_95.azy', 'et son groupe est ', 0)
('voici son fichier de correspondance:', 'obj9_160.azy', 'et son groupe est ', 0)
('voici son fichier de correspondance:', 'obj16_55.azy', 'et son groupe est ', 0)
('voici son fichier de correspondance:', 'obj18_210.azy', 'et son groupe est ', 0)
azaria@azaria-ThinkPad-X240:~/travail_indexation1$
```

FIGURE 9 – *Cluster- Kmeans*

D'où, après avoir intégrer une méthode de cluster(k-means), le temps de réaction du programme a été réduit, ce qui veut dire que k-means a augmenté le temps de réaction mais n'a pas augmenté la précision, au contraire la précision a décréu, ceci est dû au fait que nous avons seulement utilisé le descripteur de forme (Moments de Hu) comme jeu de données pour k-means, ce qui donne une faible précision de 10

À l'issue des tests que nous avons effectués et de ces différents résultats, nous avons pu remarquer que plus la valeur de  $k$  est élevée, plus nous avons une bonne classification. Plusieurs raisons expliquent les erreurs de classification. Il s'agit essentiellement des erreurs de calcul liées à la réduction du nombre de bits (de 256 à 64) pour l'échantillonnage et à la valeur de  $K$  fixée. Plus la valeur de  $K$  est grande, plus on a un bon taux de classement.

### 3 Conclusion et Perspective

En guise de conclusion, ce travail nous a permis d'avoir une idée générale sur l'indexation multimodale des contenus et de comprendre le fonctionnement des descripteurs. En effet nous avons eu à calculer les distances entre une image requête et les autres images d'une base, en utilisant les techniques d'histogrammes de couleurs, les moments de Hu et les distances globales.

Après implémentation les résultats obtenus dans la première partie étaient bons avec une précision de 100% grâce à l'utilisation de tout le descripteur par contre à la deuxième partie avec l'utilisation de k-means et la technique des moments de Hu on a remarqué que le calcul était si rapide mais la précision était si faible.

Les résultats obtenus au cours de nos expérimentations étaient valides et nous permettent de constater que certains algorithmes sont plus performants que les autres en termes d'apprentissage et que les paramètres permettant de réaliser ces apprentissages influencent les résultats.

## Références

- [1] Vector Approximation based Indexing for High-Dimensional Multimedia Databases", I. Daoudi, S. Ouatik, A. El Kharraz, K. Idrissi, D. Aboutajdine
- [2] Cours Indexation multimodale des contenus – Nicolas SIDERE – IFI 2019
- [3] [https ://docs.python.org/3/library/pickle.html](https://docs.python.org/3/library/pickle.html), Mars 2019