UNIVERSITÉ NATIONALE DU VIETNAM - HANOÏ
INSTITUT FRANCOPHONE INTERNATIONAL

OPTION : SYSTÈMES INTELLIGENTS ET MULTIMÉDIA (SIM 27)

COURS DE VISION PAR ORDINATEURS

Rapport de

Détection et Reconnaissance d'objets avec des descripteurs locaux

Groupe IV

EBWALA EBWALETTE Priscille
DIALLO Ibrahima

Professeur :
Nguyễn Thị Oanh

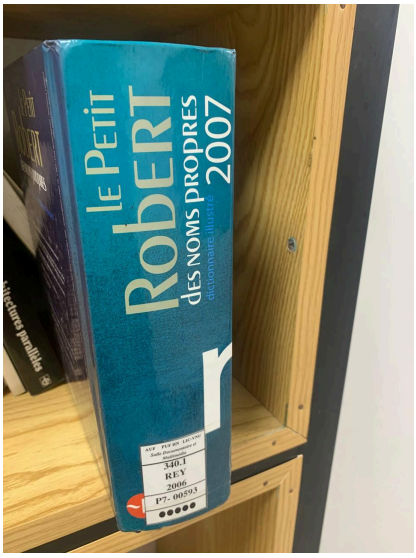
Détecteur de points d'intérêt SIFT et descripteur SIFT

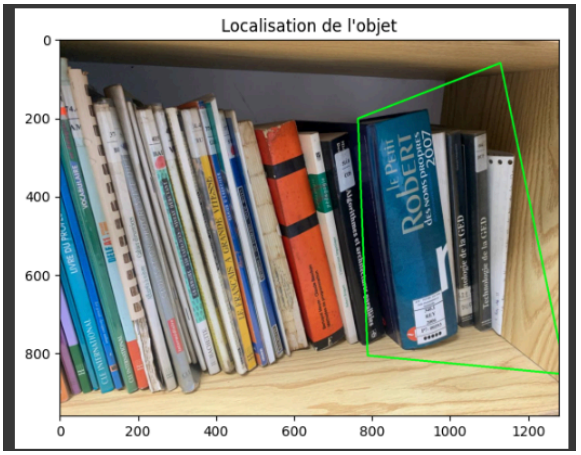
Introduction

Dans ce travail pratique, nous avons exploré la détection et la reconnaissance d'objets en utilisant des descripteurs locaux. Nous avons utilisé SIFT (Scale-Invariant Feature Transform) pour extraire des points d'intérêt et faire la mise en correspondance entre les images.
L'objectif est double :

1 Détection d'objets : Localiser un objet dans une scène contenant plusieurs objets

teste de détection d'objet et reconnaissance dans une scène:objet rechercher dictionnaire **le petit robert :**





1-1 Fonctionnement

Le programme permet de faire la reconnaissance d'objets dans une image avec le
descripteur SIFT. Cette reconnaissance nous permet de détecter un objet et de le classer dans une classe d'objet. Le programme est donc en mesure d'extraire les informations pour ensuite rentrer dans un processus pour la reconnaissance, enfin de classer l'objet dans une classe d'après les données du matching (cette section sera bien développée en deuxième partie de ce rapport).

Pour ce faire, nous avions utilisé une base d'objets contenant 7200 images qui est subdivisée en deux : 5040 données pour l'entraînement et 2160 pour le test . Sur les données d'entraînement le système extrait l'ensemble des descripteurs permettant de reconnaître les points d'intérêt sur l'image. Après la partie test une image est choisie et mise à l'entrée du système. Cette dernière est traitée dans l'optique d'extraire des descripteurs pour ensuite chercher les descripteurs à qui elle correspond le mieux parmi tous les descripteurs extrait et stocké lors de l'entraînement pour ensuite déterminer en fonction du pourcentage de matching quelle est donc la classe à laquelle appartient l'image du test.

1-2 Reconnaissance d'objets : Identifier un objet inconnu en le comparant à une base d'apprentissage.

1.2 la base utilisée :

Nous avons utilisé pour ce TP la base d'images téléchargée depuis le lien suivant :
http://www1.cs.columbia.edu/CAVE/software/softlib/coil-100.php. Cette base contient
plusieurs types d'image que nous avons récupéré 10 classe selon les instructions données. Les images ont été organisées dans des dossiers séparés pour permettre un traitement automatique,les classe utiliser sont :

- Bouteille (5 images)
- Carton (5 images)
- Voiture (5 images)
- tasse (5 images)
- pots (5 images)
- vase(5 images)
- bouteille-lait(5 images)
- Bouteille (5 images)
- Carton (5 images)
- Voiture (5 images)

1-3 Extraction des caractéristiques

Pour le travail du TP1 nous devons localiser l'objets avec SIFT extraire les caractéristiques des images qui sont calculées en un nombre restreint de pixels. Donc une zone d'intérêt (points saillants ou zone de forte contraste) est détectée et dans chacune de ces zones un vecteur caractéristique est calculé. Ce sont ces vecteurs caractéristiques qui contiennent des données importantes pour la suite de notre travail.

Parmi les méthodes permettant de détecter la zone d'intérêt et le calcul d'un vecteur caractéristique nous avions choisi le SIFT (Scale Invariant Feature Transform) au lieu du SURF (Speeded Up Robust Feature).

Pour cela, nous avions construit l'objet SIFT avec la fonction xfeatures2d.SIFT_create().

Après la fonction sift.detect() permet de trouver les points clés dans les images. Chaque point clé est une structure spéciale qui a de nombreux attributs comme ses coordonnées (x,y), la taille du voisinage significatif, l'angle qui spécifie son orientation, la réponse qui spécifie la force des points clés, etc.

1-4 Création du tableau numérique pour la liste des points clés.

Ici, nous avons créé un tableau numérique, des keypoints et des descripteurs de formes. La librairie Open CV nous offre plusieurs alternatives pour cela, mais nous avons utilisé la fonction sift.detectAndCompute() qui permet de rechercher directement les points clés et les descripteurs en une seule étape.

cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS

nous retourne des cercles, la taille et les orientations des points clés de nos images. Ayant donc les points clés et les descripteurs nous allons pouvoir aller à la seconde partie de notre programme. Cette partie consiste à faire correspondre les points clés des images tests à ceux que nous avons eu lors de l'entraînement.

2. Reconnaissance d'Objet

La reconnaissance est le résultat de deux parties de notre programme : la mise en correspondance (matching) et le calcul de nombre de correspondances entre les images.

2-1 La mise en correspondance

Nous avons pour chaque image prise en entrée déterminée les points clés trouvés, ensuite nous avons essayé de faire correspondre les caractéristiques de l'image à ceux déjà calculés pour les images de training. Notons que les images d'entraînement sont déjà regroupées en classes.
Ainsi donc, nous avons utilisé la technique d'appariement croisée pour calculer les distances entre les points. Dans cette technique nous avons

choisi un seuil 0,6 ensuite, nous avons calculé la distance absolue et enfin nous avons classé par ordre les ratios entre le plus proche et le deuxième plus proche. Les ratios supérieurs au seuil sont retenus.

Nous avons ensuite implémenté la mise en correspondance en utilisant la fonction Brute force Match de OpenCV avec notre calcul de distances déterminé précédemment. Nous rappelons que le seuil défini nous permet d'éliminer les fausses correspondances.

Pour déterminer les fausses correspondances, nous avons utilisé la technique donnée qui est de calculer le ratio entre la distance du descripteur de l'image inconnue et le descripteur de l'image modèle. Si ce ratio est inférieur à un seuil= 0.6 , alors la correspondance peut être considérée comme robuste, sinon elle est rejetée

3. implémentation.

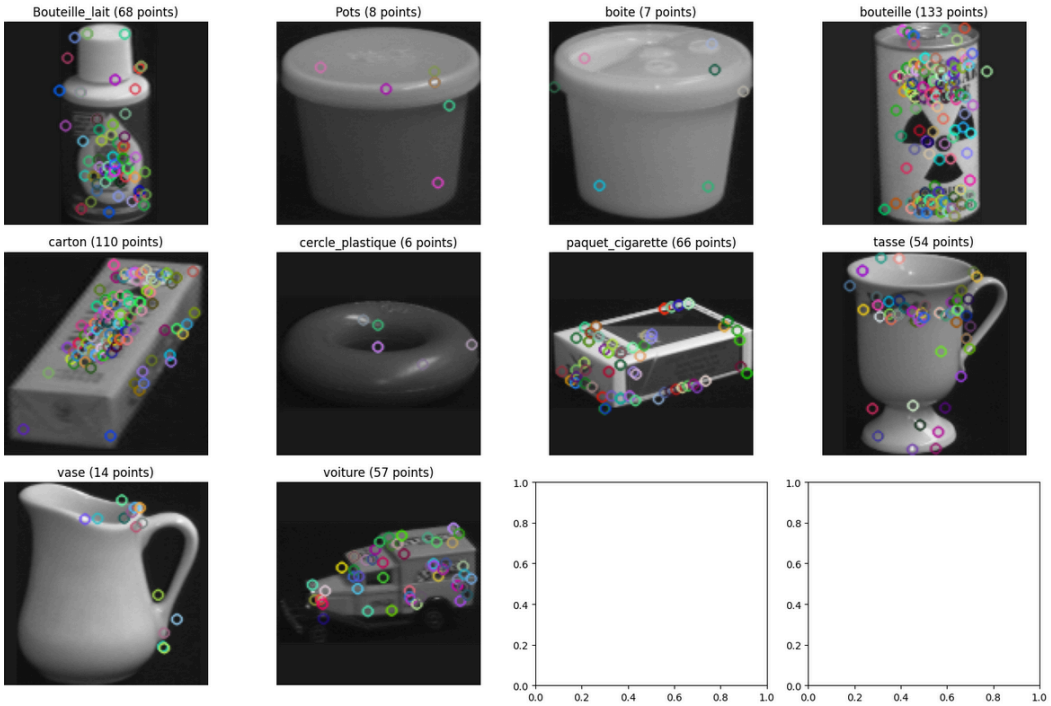
Dans cette étape nous allons détailler notre implémentation. Nous avons travaillé sur Google colab, pour avoir une vitesse de traitement beaucoup plus grande. Notre notebook est tp1vod_detection.ipynb Le dossier récupéré, nous avons créé un sous dossier contenant nos 10 classes sur drive .Voici un aperçu des différentes classes :

3-1 Le dataset

Nous avons utilisé le dataset fourni dans le questionnaire Nous avons utilisé la fonction train_test_split de la bibliothèque scikit learn pour spliter notre dataset de 7200 images en 70% pour l'entraînement et 30% pour le test. Nous avons précisé la variable random_state=42 pour persister les données divisées, afin d'avoir les mêmes données en entrainement et test à chaque réexécution.

3.2. Calcul et affichage des points d'intérêts sur les images

Nous avons utilisé notre fonction calculateDescriptors pour calculer les descripteurs locaux pour chaque image, une boucle étant faite sur la totalité des images.
Après ce calcul l'affichage donne le résultat suivant (quelques exemples étant pris au hasard) :

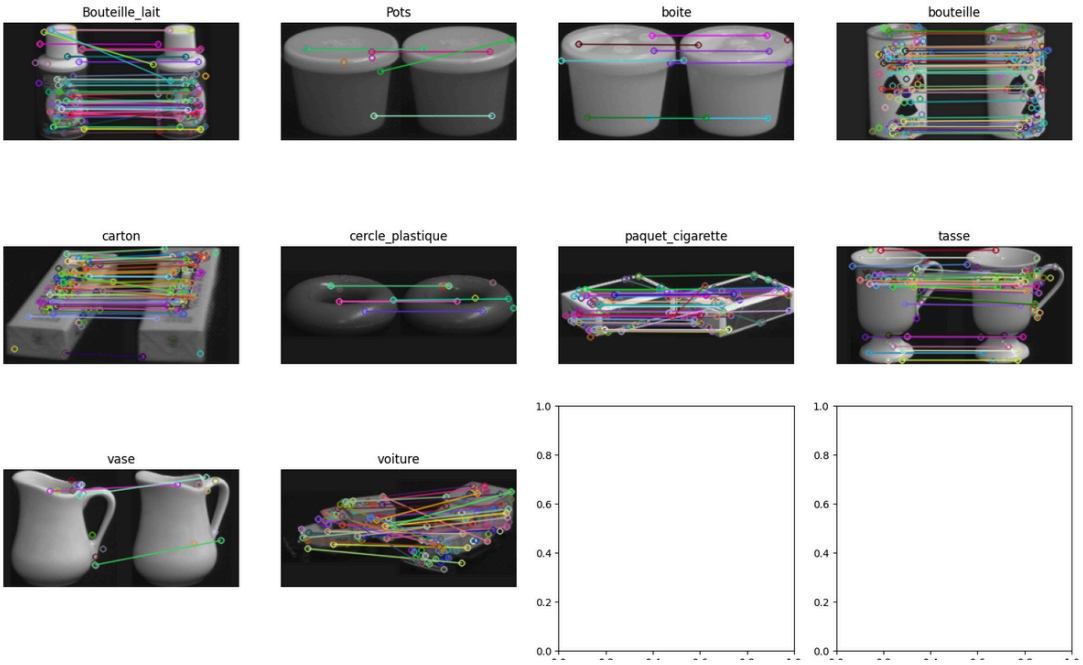


Nous sauvegardons après les images avec les descripteurs et les points d'intérêts dans des fichiers au format npy dans le but de ne plus refaire toutes ces étapes lors d'une prochaine exécution.

3.3. Correspondance ou matching entre les images

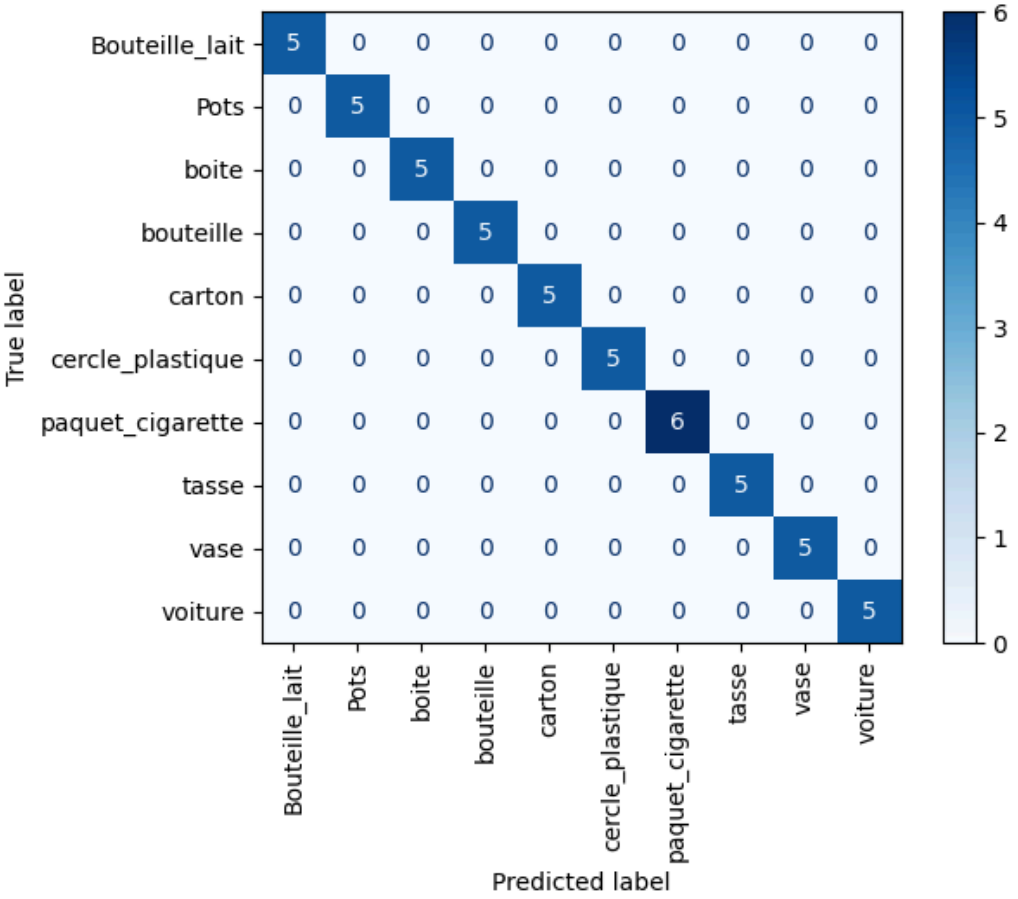
A cette étape du travail il est question de prendre une image en test et comparer ses descripteurs avec tous les descripteurs des classes déjà sauvegardés et d'en faire un tri par ordre décroissant de nombre de correspondances. La classe qui aura le plus grand score de correspondance sera la classe prédite.
Pour bien visualiser nos résultats nous avons utilisé la fonction drawMatches de cv2 pour relier les similarités.

Le résultat de cette étape nous donne :



4.Evaluation des résultats

Nous avons remarqué que la prédiction est bonne sur toutes les images comme le montre cette image ainsi, Pour l'évaluation du modèle, nous avons utilisé la matrice de confusion suivante.



Conclusion

Ce TP nous a permis de mettre en pratique les notions de détecteurs de points d'intérêts appris au cours. Nous avons compris que pour effectuer une correspondance entre deux images, nous pouvons passer par les points d'intérêts. Les descripteurs permettent de faire une reconnaissance entre les objets. Les objectifs fixés au début de ce TP ont été atteints, mais nous prévoyons comme perspective ajouter d'autres méthodes de détection et de reconnaissance d'objets à notre programme afin de pouvoir comparer les résultats. Nous pourrions également travailler avec d'autres jeux de données ayant beaucoup plus d'images pour améliorer nos résultats.