RAPPORT TRAVAUX PRATIQUES

Reconnaissance d'objets avec SIFT

Mamadou Ben H Cissoko - Djeila Diakite

November 13, 2019

Professeur :
DR NGUYEN THI OANH

1 INTRODUCTION

De nos jours , vision par ordinateur est un domaine scientifique interdisciplinaire qui traite de la façon dont les ordinateurs peuvent être conçus pour acquérir une compréhension de haut niveau à partir d'images ou de vidéos numériques . Du point de vue de l' ingénierie , il cherche à automatiser les tâches que le système visuel humain peut effectuer. La discipline scientifique de la vision par ordinateur s'intéresse à la théorie derrière les systèmes artificiels qui extraient des informations à partir d'images,dont les résultats sont plus en plus optimal. Dans notre premier TP on a utilisé une technique traditionnelle qui consiste à utiliser les histogrammes des pixels créés à base d'une base d'image et utiliser utilisant le théorème de Bayes afin de prédire la classe d'appartenant des nouveaux pixels d'images inconnus. Certes, cette dernière peut déjà être utilisé pour divers problème tels que la détection de peau, détection des régions d'objets, etc. Mais cela avec une performance très sensible et peut être affecté par la qualité des images de la base et en entrée, et aussi l'homogénéité de ces dernières. Nous allons utiliser maintenant le descripteur dans la reconnaissance d'objet.

2 PRÉSENTATION DE L'APPLICATION

Pour exécuter notre projet :
Étape 1 : Il faut installer les librairies nécessaires;
Étape 2 : A partir du terminal ,Taper **python reconnaissance.py** et d'autres fichiers sont présents pour l'apprentissage tels que: **Split.py** permettant de diviser la base des images en deux parties avec un pourcentage de 50 pourcentage pour la base d'entraînement et 50 pourcentage pour la base de test et le fichier pour la **confusion matrix.py** exporter des résultats de nos descripteurs exportés sous format textes.
Pour utiliser l'application, il suffit juste de lancer la reconnaissance en donnant en paramètre une image.

2.1 PRÉSENTATION DU JEU DE DONNÉES

Afin de tester la robustesse de cette approche et voir plus de résultat à interpréter, on a utilisé deux jeux de données :
1 - Base des images (données 1) :

Données très variées, dans lesquelles, les images de la même classe sont très différentes.
les unes des autres au niveau de la taille couleur, forme, etc. Ce sont des différences naturelles et vue la nature de ces images et la position à laquelle elles ont prises peuvent rendre la correspondance entre les images plus difficile car les images ont prises dans des circonstances différentes . Ce sont les images de **Columbia University Image Library (COIL-100) [processed]**
2 - Base des images (données 2) :
Données plus ou moins homogènes, dans lesquelles seules différences entre les images de la même classe, sont fait d'une manière manuelle en tournant, rognant, zoomant,le contraste des ces images est grande et en grande ayant des bruits. **Caltech 101 [101 Object Catégories]**

3 PRÉPARATION DE LA BASE D'APPRENTISSAGE

Pour préparer notre base d'apprentissage, on a d'abord créer une fonction permettant de diviser la base des images en 2 parties, une pour l'entraînement et une pour le test. Il y a des classes qui ont plus d'image, donc on normalisé la taille des données dans chaque dossier. Normalement on doit faire des expérimentations avec des différentes taille de sectionnement mais comme ce n'était pas le but de ce TP d'étudier le partitionnement des données d'apprentissages et test a été fixé à 50% pour l'entraînement et 50% pour le test, et deux dossiers **training et test** seront crées dans les quels chaque classe d'image sera créée avec avec le partitionnement des images dans leurs classes respectives.

- **classe BACKGROUND-Google :** 56 images de test et 58 de training (Base 1)
- **classe Object :** 20 images de test et 52 de training (Base 2)

4 EXTRACTION DES CARACTÉRISTIQUES DES IMAGES D'ENTRAÎNEMENT

Dans cet étape on a créé notre base de référence de descripteur. Les étapes sont les suivantes:
1. Détecter les points d'intérêts avec SIFT.
2. Récupérer les descripteurs associés à chaque image de notre base : Pour faire cela on a utilisé une boucle qui nous permettra de parcourir chaque classe de notre base d'entraînement et on extrait les caractéristiques (point d'intérêt et le descripteur de chaque image).
3. Stocker les descripteurs et les noms de classes de chaque image dans un tableau qu'on utilisera après pour faire la correspondance entre l'image de teste et la classe de l'image mais aussi pour la matrice de confusion.
4. Sérialisation dans un seul fichier du tableau des descripteurs et des classes des images. On fait la même chose pour chacune des images du dossier train de chaque classe.

4.1 DÉTECTION DES POINTS D'INTÉRÊTS

Dans **OpenCv** , il y a beaucoup de fonction qu'on peut utilisés pour détecter les points d'intérêt au sein de l'image. Par exemple **SURF, SIFT , HARRIS CORNER DÉTECTION , FAST, BRIEF, et ORB**,mais dans ce TP nous avons avons utilisés SIFT .On crée l'objet SIFT et peut prendre en paramètre le nombre de point maximale à détecter. Parfois des images ont plus de point d'intérêt et d'autre moins, et ça peut jouer vraiment sur le temps de traitement de toute la base.Pour avoir normaliser ça on a fixé le nombre de point détecté pour chaque image.

4.2 RÉCUPÉRER LES DESCRIPTEURS ASSOCIÉS À CHAQUE IMAGE:

Les descripteurs s'obtiennent avec avec les points en faisant le **detectAndCompute** dans**Opencv**. On les stocke dans un tableau à double dimension [classe][image]. Exemple : pour 10 classes avec 35 images d'entraînement on aura 100x35 = 3500.

4.3 SAUVEGARDE DES DONNÉES

Pour stocker nos données d'entraînement, on a utilisé la stérilisation pour conserver les états de notre tableau.

5 TEST

Pour tester, on parcourt le dossier test de chaque classe, et pour chaque image on fait : 1. Récupère ses points d'intérêt.
2. Pour fait le Matching (correspondance) de l'image de test avec chacune image dans la base , On trie les images de la base d'entraînement avec le nombre de correspondance de point d'intérêt et puis en fonction du nombre de voisin le plus proche qu'on veut avoir on prends les n premiers éléments du tableau trié avec la valeur de **K (les voisins les plus proches)**.
3. On prend la classe dominante comme la classe de l'objet en entrée ayant la distance euclidienne la plus petite par rapport a l'image de teste.

5.1 MATCHING AVEC CHACUNE DES IMAGES DANS LA BASE

Le matching consiste à comparer les points d'intérêt entre deux images données. Pour affirmer qu'un point d'intérêt X d'une image A et correspond à un point dans une autre image B, on calcule le carré de distance entre ce point (de A) avec tous les point dans B et on trie puis prends 2 premiers points Y et Z qui ont le carré de distance plus petit. Si la différence entre XYet XZ est inférieur à 0.7 on dit que celui qui a la plus petite distance avec X est le point correspondant de X dans dans B, sinon il n'y a pas de correspondance. Pour faire ça dans **Opencv** il existe plusieurs algorithmes de KNN implémenté comme le **Flann KNN, BrutForce, Best-**

Matcher, et on n'a pas testé plusieurs algorithmes de ce type mais on a directement utilisé le **FLANN based Matcher**.

5.2 Création du tableau trié sur le nombre de correspondance

A chaque image trouvée dans les dossiers tests on essaye de faire la correspondance et on compte le nombre de correspondance avec chaque image, et on trie à base de celle-ci pour avoir un tableau de plus grand vers le plus petit. On peut prendre après les premiers voisins le plus proche de l'image de test car on a définie notre **K = 5** pour la première base.

5.3 Choix de la classe d'appartenance:

Pour le choix de la classe d'appartenance de l'objet de test on prend dans dans le tableau des **n voisins** proches de l'objet et dans ce tableau on prend la classe plus répétée de ce tableau. Donc cette classe correspond alors à la classe d'appartenance.

5.4 Matrice de confusion :

la matrice de confusion est une matrice qui mesure la qualité d'un système de classification. Chaque ligne correspond à une classe réelle, chaque colonne correspond à une classe estimée. La cellule ligne L, colonne C contient le nombre d'éléments de la classe réelle L qui ont été estimés comme appartenant à la classe C, pour avoir cette matrice on a créé un tableau **A** de taille [nbclasse][nbclasse] double dimension avec la taille de nombre de classe de chaque image . On initialise ses valeurs à 0 au début du test. Après chaque image on prend la classe de l'image qui est c et la classe prédite c1. On incrémente la valeur à l'emplacement A[c][c1] +=1. Pour visualiser notre matrice pour nos différentes expérimentations, on transforme ce tableau en une matrice et désignant chaque valeur par une couleur en fonction du nombre de correspondance.

5.5 Évaluation du modèle

Pour évaluer notre modèle de prédiction on a utilisé le pourcentage des images classifiées correctement sur le nombre totale des images.
Taux de précision = Nombre de prédiction correcte / Nombre totale des images.

6 Expérimentations :

Pour nos expérimentations, on fait varier beaucoup de paramètre:

- **La valeur des paramètres de FLANN :** du FLANN based Matcher car la distance euclidienne est très importante pour déterminer l'appartenance des objects car plus la

distance est longue plus l'objet est loin d'avoir des points similaires avec l'objet de reconnaissance donc de ce fait nous avons beaucoup ajusté le paramètre afin d'avoir plus de précision. 56 images de test et 58 de training (Base 1)

- **La valeur de K:** Le nombre de voisin le plus proche lors de prédiction, symbole k. comme on a pas la valeur de paramètre la plus adapté et laquelle montre plus de bon résultat donc il était nécessaire d' ajuster afin d'avoir plus de bons résultats.
- **Matrice de confusion:** pour les différentes classes de modèle pendant l'expérimentation Et on affiche la matrice de confusion pour le paramètre qui donne le meilleurs résultat pour chaque base.

6.1 Expérimentations:

6.1.1 Base des images 1:

Descripteurs	K valeurs	FLANN value	Precision du modele
100	5	0.6	12.73
200	3	0.6	18.57

6.1.2 Base des images 2:

Descripteurs	K valeurs	FLANN value	Precision du modele
100	5	0.6	61.0 %
150	3	0.6	75 %

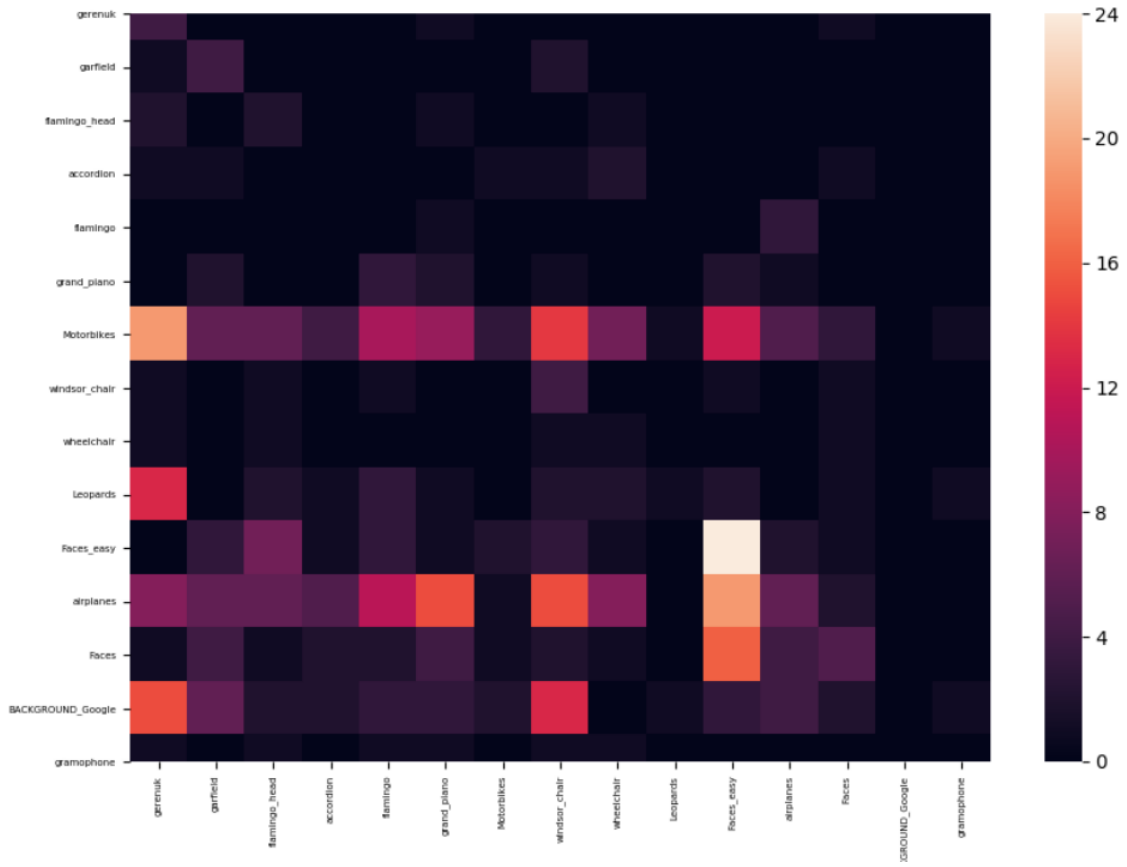
7 INTERPRÉTATION ET ANALYSE DES RÉSULTATS

Parmi les multiples expérimentations qu'on a effectué avec differents paramètres on a retient les differents meilleurs résultats donc les deux meilleurs:

7.1 BASE 1:

Descripteur	K valeurs	FLANN value	Precision du modele
200	3	0.6	18.57 %

la matrice de confusion:



Pour la base 1 le modèle a eu une précision de **18.57** comme de taux correction au maximum, cet résultat est normal vue les images viennent sont prises dans des conditions différentes ou proviennent des sources différentes. Mais aussi due aux différences naturelles des images, c'est-à-dire les différences entre les images ne sont pas dues uniquement qu'aux transformations manuelles telles que les rognages, rotations, zoom, le contraste est diffèrent pour les images , des tailles et des formes différentes ce qui a impacté notre modèle a avoir un taux de précision faible malgré une grande quantité des images de base pour l'entraînement et de plus il classifie mal les images certaines classes malgré que ces classes contiennent certaines images de bonne qualité, comme montre la matrice de confusion dans le résultat de l'expérimentation . Il y a des images qu'il arrive à bien classifié comme le montre notre matrice de confusion.

Figure 7.1 classe face - classe face-eyes

On a essayé de voir sur la matrice de plus près les images qui ont une forte similarité (confusion) entre eux et on a remarqué que vraiment ces images ont une ressemblance au niveau de la forme par exemple la classe **classe face - classe face-eyes** (images ci-dessous) il y a une forte confusion entre eux et l'extraction des caractéristiques sur différentes classes permet de classifier les classes des images, la forme , la position des visages sur les deux classes est bonne et nette d'où il est parvenu a trouver une corrélation forte entre les deux. Avec SIFT malgré le fait de trouver beaucoup des points , la classe de correspondance est seule où on trouve le max de taux de correspondance

le taux de Précision va être :

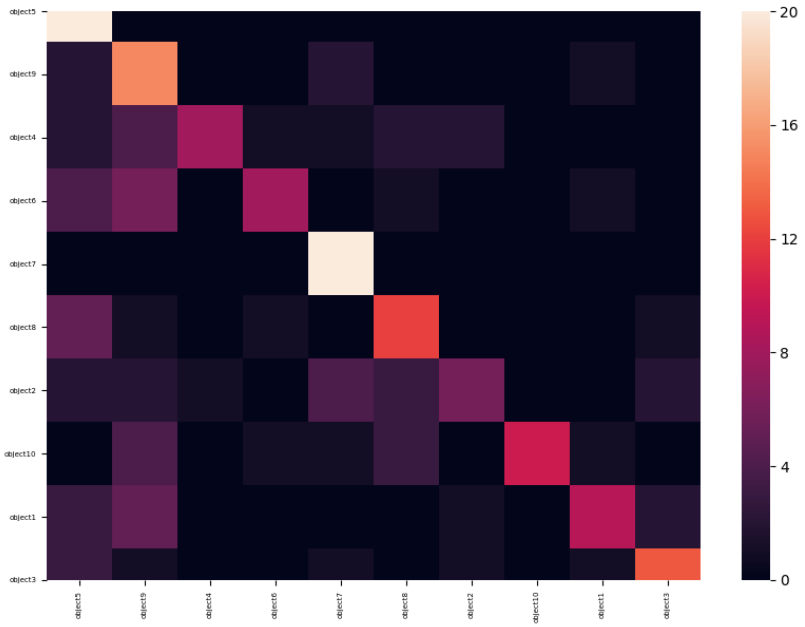
Taux = Y/X d'où le résultat décroît lorsqu'on prend plus de descripteur. La variation de k voisins dans cette base affecte aussi le taux de correction puisqu'on voit que plus on augmente le nombre de voisin plus ça descend puisque le point de correspondance est très peu et ça ne laisse pas beaucoup de différence entre la bonne et la mauvaise classe pourtant on prend la classe dominante dans les voisins. Par exemple :

Classe	obj1	obj1	obj1	obj2	obj3	obj3	obj2	obj2	obj2
correspondance	2	1	1	1	1	1	1	1	1

Si on va déduire la bonne classe par la classe dominante des différentes classes dans le voisinage dans ce cas la bonne classe c'est le **obj1** .

7.2 BASE 2:

Pour la base 2, nos résultats sont à 75% de taux précision au maximum, cela est vraiment normale car les images au sein de la même classe sont déformées par exprès, avec les rognages, rotations, agrandissement, réduction, ajout des bruits (comme expliqué par le propriétaire du dataset. Et d'où l'image dans le dossier d'entraînement et le test sont les mêmes images mais rajouté des modifications très légères. Et on a aussi une matrice de confusion assez bien claire sur laquelle on a presque la diagonale en couleur très claire ce qui signifie un meilleur taux de correction et de correspondance entre les classes des images ayant de similarité.



Exemple d'un base 2 :

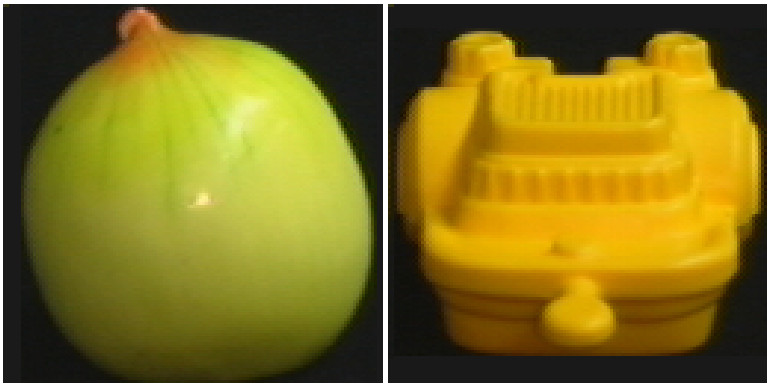


Figure 7.2 classe obj2 - classe obj3

En observant la matrice de confusion d'entre de ces deux images, On aperçoit une légère correspondance entre ces deux classes car, si on le met à niveau de gris ou si on applique un flou de ces deux, elles deviennent de plus en plus indiscernables. On sait que SIFT dans sa robustesse, n'est pas sensible à des transformations courantes, mais dans le cas ou on a n'a pas assez de variété au sein de notre classes (comme le cas de la base 1) ces genres d'erreur (confusion) n'est pas inévitable.Ce qui nous mène à définir la raison de cette confusion, c'est le manque de variété dans chaque classe de notre base d'images. Ce qui nous mène à définir la raison de cette confusion, c'est le manque de variété dans chaque classe de notre dataset. Le résultat de celle-ci est plutôt contrairement au celui de la première base car on a assez de correspondance des points d'intérêts et ce qui nous résulte ici que le maximum de taux de correspondance a été trouvé avec le plus grand nombre de descripteur.

7.3 Conclusion

Dans ce TP nous a permis de faire une application qui peut reconnaître les objets dans une image en utilisant le descripteur SIFT, donc ça prend entrée une image et donne en sortie le nom de l'objet contenu dans l'image. En sachant que cela a été faite en utilisant une base d'image, et afin de voir l'efficacité de SIFT on a utilisé deux bases qui sont complètement différentes car elles viennent de sources différentes. Une première qui contient des images totalement variées de même type, et une autre contient des images de même classes très homogène. D'après nos résultats, on peut conclure que SIFT est une méthode très robuste pour la reconnaissance d'objet. mais présentant des limites, qui le plus souvent est le temps d'extraction des points d'intérêts au sein d'une image au cas ou on ne limite pas le nombre.

Plus de descripteur c'est plus lent mais augmente la précision et moins de descripteur plus rapide mais affecte plus ou moins le résultat et il y a un très fort décalage comme on a vu dans nos graphiques et matrice de confusion. Au vue de nos différentes expérimentations on a vu qu'elle donne plus de meilleurs résultats pour les images modifiés manuellement qui sont de même type. Elle donne un résultat moins bon pour des images issues des sources différentes c'est a dire sur des images de même type d'objet mais acquise d'une manière, formes, tailles, traitements très différents .

7.4 RÉFÉRENCES

[1] Feature Matching opencv https://docs.opencv.org/3.4/dc/dc3/tutorial-py-matcher.html
[2] Dataset (base 1) http://www1.cs.columbia.edu/CAVE/software/softlib/coil-100.php [3] Dataset (base 1) http://www.vision.caltech.edu/Image-Datasets/Caltech101/

REFERENCES