ENSTA-Paris 2e année - Cours MI204
TP1 : Détection et Appariement de Points Caractéristiques

L'objectif de ce TP est de (1) se familiariser avec la bibliothèque de traitement d'images OpenCV sous Python, et (2) expérimenter, critiquer et mettre en œuvre différentes techniques de *représentation* pour l'appariement de structures locales dans les images, en suivant les différentes étapes de la problématique :

- DÉTECTEUR : Comment réduire le support de la représentation.
- DESCRIPTEUR : Quelle information attacher à chaque point du support.
- MÉTRIQUE : Quelle mesure utiliser pour apparier des points.
- RECHERCHE : Comment coder et parcourir l'espace des points d'un modèle.

Vous devez fournir un rapport, au format pdf, en justifiant autant que possible vos réponses avec des expériences personnelles.

1 Préliminaires informatiques

Une base de code est fournie, qui fonctionne avec une installation standard d'OpenCV (sans opencv-contrib, version testée 4.1.0) et Python3. L'archive contenant cette base peut être téléchargée au lien suivant :
`https://perso.ensta-paris.fr/~manzaner/Cours/MI204/TP1_Features.zip`

Tous les codes fournis sont des scripts Python3 éditables et exécutables en lançant la commande, par exemple :
```
$ python3 Convolutions.py
```

Certains scripts nécessitent des arguments, voir le code Python pour plus de détails.

On trouvera quelques paires d'images de test dans le même répertoire que le logiciel de TP :
`http://perso.ensta-paris.fr/~manzaner/Cours/MI204/Image_Pairs.zip`

OpenCV est une bibliothèque très utilisée en analyse d'images et on pourra consulter avec profit de nombreux guides et tutoriels en ligne (attention toutefois aux versions de Python et d'OpenCV fournies dans les exemples). On se limitera ici à mentionner les sites officiels pour le guide de référence des fonctions :
`https://docs.opencv.org/4.1.0/`
ainsi que le tutoriel :
`https://docs.opencv.org/4.1.0/d6/d00/tutorial_py_root.html`

2 Format d'images et Convolutions

Q1 Expérimenter le code de convolution fourni en exemple dans *Convolutions.py*. Observer la différence entre le calcul direct par balayage du tableau 2d et le calcul utilisant la fonction *filter2d* d'OpenCV. Déchiffrer les fonctions OpenCV utilisées pour la lecture et la copie d'images, ainsi que la fonction MatPlotLib utilisée pour l'affichage.

Q2 Expliquer pourquoi le noyau de convolution fourni en exemple réalise un réhaussement de contraste par rapport à l'image originale.

Q3 Modifier le code pour calculer les convolutions qui approximent les composantes du gradient $I_x = \frac{\partial I}{\partial x}$ et $I_y = \frac{\partial I}{\partial y}$. Calculer ensuite la norme euclidienne du gradient $||\nabla I|| = \sqrt{I_x^2 + I_y^2}$. Quelles précautions doivent être prises pour obtenir un affichage correct (i.e. qui montrent l'ensemble des valeurs, négatives comme positives) dans tous les cas ?

3 Détecteurs

Q4 Compléter le code fourni dans le script *Harris.py* pour calculer la fonction d'intérêt de Harris (à une seule échelle, en utilisant une fenêtre **W** de taille fixe), et les points d'intérêt correspondants. Expliquer comment le code fourni, qui utilise la dilatation morphologique (maximum dans un voisinage donné), permet de calculer les maxima locaux de la fonction d'intérêt *Theta*.

Q5 Commenter les résultats obtenus avec votre détecteur de Harris et l'effet des paramètres utilisés, taille de la fenêtre de sommation et valeur de α en particulier. Comment peut-on réaliser ce calcul sur plusieurs échelles ? Comment étendre la notion de maxima locaux pour faire en sorte que deux points d'intérêt soient toujours distants d'au moins r pixels ?

Q6 Expérimenter et comparer les deux détecteurs ORB et KAZE en lançant le script *Features_Detect.py*. Rappeler le principe de chacun de ces détecteurs. Expliquer les principaux paramètres propres à chaque détecteur et leur effet sur la détection. Comment peut-on visuellement évaluer la répétabilité de chaque détecteur appliqué sur une paire d'images ?

4 Descripteurs et Appariement

Q7 Expliquer le principe des descripteurs attachés aux points ORB et ceux attachés aux points KAZE. Quelles propriétés des détecteurs et/ou des descripteurs (distinguer les deux aspects dans la réponse), permettent de rendre l'appariement invariant par changement d'échelles et invariant par rotation ?

Q8 Expliquer et comparer qualitativement les performances des trois stratégies d'appariement de points d'intérêt réalisées dans les trois scripts *Features_Match_CrossCheck.py*, *Features_Match_RatioTest.py* et *Features_Match_FLANN.py*. Expliquer pourquoi les distances utilisées sont différentes pour les deux descripteurs.

Q9 Proposer une stratégie pour évaluer quantitativement la qualité des appariements en déformant une image avec une transformation géométrique connue (on pourra utiliser une fonction telle que *cv2.warpAffine*).