TP4 - Hiver 2026

IMN 259

Analyse d'images

Date limite pour remettre votre travail : 19 avril à 23h59

Objectifs

Implémenter des opérations ponctuelles et extraire des caractéristiques

1. Détection de contours
2. Détection de coins
3. Segmentation

Description

À l'aide du code Python fourni (fichiers *tp4.py, MI_image_numpy.py*) vous devez implémenter différentes fonctions vues dans le cours. Pour ce faire, il est fortement recommandé de récupérer les fonctions *EdgeDetec*, *NonMaxSupp*, *ZeroCrossing*, *HarrisCornerDetec*, et *KMeansSegmentation* du fichier *MI_image_numpy.py*. Toutefois, vous êtes libre d'ajouter d'autres fonctions si vous en éprouvez le besoin. Évidemment, il est fortement recommandé de récupérer du code des travaux pratiques 1, 2 et 3.

Recommandations :

1. tp4A : **Seuil de la norme du gradient**. Pour ce numéro, vous devez implémenter l'algorithme du «seuil du gradient»présenté dans le chapitre sur l'extraction de caractéristiques. Testez votre approche minimalement avec les images : *form, shapes, cameramanNoise*, et *clown*.
2. tp4B : **Zero-Crossing**. Pour ce numéro, vous devez implémenter l'algorithme du «zero-crossing »présenté au chapitre sur l'extraction de caractéristiques. Toutefois, il vous faut ici remplacer les 0 par le seuil passé en paramètre (variable *threshold*). Vous devez également implémenter le filtre laplacien *LaplacianFilter* ainsi qu'utilisé un filtre passe bas (gaussien ou moyenneur de votre TP3). Testez votre approche minimalement avec les images : *form, shapes, cameramanNoise*, et *clown*.
3. tp4C : ***Suppression des non-max**. Vous devez implémenter l'algorithme de *thinning* présenté au chapitre sur l'extraction de caractéristiques. Vous n'avez pas à appliquer de filtre passe bas (gaussien ou autre). Testez votre approche minimalement avec les images : *form, shapes, cameramanNoise*, et *clown*.
4. tp4D : ***Détecteur de Harris**. Vous devez implémenter l'algorithme suivant :
   (a) Calculer la dérivée en X de l'image d'entrée $f(x, y)$: $f_x = \partial f / \partial x$.
   (b) Calculer la dérivée en Y de l'image d'entrée $f(x, y)$: $f_y = \partial f / \partial y$.

(c) Pour chaque pixel (x, y), calculer les valeurs A, B et C de la matrice hessienne

$$\begin{aligned} A &= \sum_{i,j} W_{i,j} f_x(i,j)^2 \\ B &= \sum_{i,j} W_{i,j} f_x(i,j) f_y(i,j) \\ C &= \sum_{i,j} W_{i,j} f_y(i,j)^2 \end{aligned}$$

où $W_{i,j}$ est un coefficient gaussien.

(d) Calculer $M_c(x, y) = det(H) - k.trace(H)^2$.

(e) Lorsque tous les pixels auront été visités, ramener la dynamique de l'image M_c entre 0 et 255 (fonction Rescale).

(f) Sauvegarder M_c

Testez votre approche minimalement avec les images : *form, shapes*, et *cameramanNoise*.

5. tp4E : Segmentation par **K-moyennes**. Vous devez implémenter l'algorithme des K-moyennes présenté au chapitre sur l'extraction de caractéristiques, pour $K = 2$ classes seulement. Testez votre approche minimalement sur les images : *form, shapes* et *satellitelles*. Si vous avez du temps, implémentez une version K-moyennes pour un K arbitraire.

Afin de vous simplifier la vie, vous n'avez qu'à traiter des **images en niveau de gris** pour ce TP.

(*) Problèmes un peu plus difficiles que je vous recommande de faire en dernier.

Évaluation

Ce travail doit être fait en **équipe de UN ou DEUX**. Remettez un rapport simple expliquant vos solutions. Au moment de soumettre votre travail, assurez-vous que votre code roule bien et que tous les fichiers nécessaires sont soumis. Si vous installez des librairies, svp fournir le fichier requirements.txt.

IMPORTANT

Si vous utilisez l'IA générative ou autres sources (*Google, StackOverflow*, etc.) pour vos solutions, vous devez les citer dans votre code. Je me réserve de vous demander à l'oral de m'expliquer le contenu de vos codes à tout moment après la remise.