TP 1 – De l'image aux descripteurs

Tous les exercices des TP devront donner lieu à un descriptif/commentaire qui sera intégré dans le compte-rendu des TP à remettre après la fin des TP pour évaluation. Ce rapport sera inclus dans un fichier ZIP contenant les codes Python produits pour chaque exercice (un par binôme).

Exercice 1 : Prise en main de Python + OpenCv

Editer le fichier videoprocessing.py avec le code python ci-dessous. Ajoutez les commentaires manquants.

```
# Mon script OpenCV : Video_processing
#
import numpy as np
import cv2

#
def imgproc(imgc):
        return imgc

#
cap = cv2.VideoCapture('data/jurassicworld.mp4')

#
while (True):

    #
    ret, frame = cap.read()

    #
    if ret == True:
        #
        img = frame.copy()
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

        #
        gray = frame_processing(gray)

        #
        cv2.imshow('MavideoAvant', frame)
        cv2.imshow('MavideoApres', img)

    else:
        print('video ended')
        break

    if cv2.waitKey(1000) & 0xFF == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

Exercice 2 : Filtrage, convolution, détection de contours

Question 1 :

Complétez le code fourni afin d'appliquer différents traitements de filtrage (par exemple flou gaussien, masque median, …) à la frame courante avant affichage. Insérez certains résultats dans votre rapport et commentez-les afin de démontrer que vous avez compris le mécanisme de filtrage d'images

https://docs.opencv.org/3.2.0/d4/d13/tutorial_py_filtering.html

Question 2 :

Complétez le code fourni afin d'appliquer différents algorithmes de détection de contours (par exemple Sobel, Laplacien, Canny …) à la frame courante avant affichage. Insérez certains résultats dans votre rapport et commentez-les afin de démontrer que vous avez compris le mécanisme de filtrage d'images.

https://docs.opencv.org/3.2.0/d5/d0f/tutorial_py_gradients.html
https://docs.opencv.org/3.2.0/da/d22/tutorial_py_canny.html
https://docs.opencv.org/3.2.0/d3/d05/tutorial_py_table_of_contents_contours.html

Exercice 2 : Images binaires et opérations entre images

Question 1 :

Complétez le code fourni afin d'appliquer différents algorithmes de seuillage / binarisation (seuillage fixe ou adaptatif) suivi d'opérations de morphologie mathématique (érosion, dilatation) à la frame courante avant affichage. Insérez certains résultats dans votre rapport et commentez-les afin de démontrer que vous avez compris le mécanisme de filtrage d'images

https://docs.opencv.org/3.2.0/d7/d4d/tutorial_py_thresholding.html
https://docs.opencv.org/3.2.0/d9/d61/tutorial_py_morphological_ops.html

Question 2 :

Complétez le code fourni afin d'appliquer une opération de soustraction entre 2 frames successives avant affichage. Expliquez l'intérêt de ce type de traitement (opérations entre frames successives) ?

https://docs.opencv.org/3.2.0/d0/d86/tutorial_py_image_arithmetics.html

Exercice 3 : Détection de changement de plan et résumé automatique de vidéo

Question 1 : Changement de plan

Exploitez certaines des méthodes vues précédemment pour mettre en place un mécanisme de détection automatique de changement de plan dans les vidéos. Testez et commentez ses performances. Proposez sans le coder un mécanisme que vous jugeriez plus performant.

Question 2 : Création d'Images résumés de plan

Exploiter l'ensemble des méthodes vues précédemment pour mettre en place un mécanisme de génération automatique de résumés de plans dans les vidéos. Testez et commentez ses performances. Proposez sans le coder un mécanisme que vous jugeriez plus performant.

Exercice 4 : Calcul de descripteurs SIFT, appariement de descripteurs et résumé automatique de vidéo

Question 1 : Construction de la liste d'images d'un même plan

A l'aide du mécanisme pour la détection automatique de changement de plan de l'Exercice 3, mettez en place un algorithme qui construit une liste stockant l'ensemble des 'images correspondant à un seul plan. Testez et commentez ses performances. Proposez sans le coder un mécanisme que vous jugeriez plus performant.

Question 2 : Calcul de descripteurs SIFT et appariement

Créer une fonction qui, étant donnée deux images, calcule les points SIFT des deux images et trouve les correspondances entre ces points (provenant des deux images). Ajouter un paramètre *seuil* à la fonction pour retenir que les meilleurs correspondances (celles dont la distance est inférieure au seuil fourni). Testez et commentez ses performances. Proposez sans le coder un mécanisme que vous jugeriez plus performant.

Question 3 : Recherche du meilleur appariement

A l'aide de la fonction précèdente, créer une fonction qui, étant donnée une séquence d'images, fournit l'image qui a le meilleur appariement avec sa précédente dans la séquence. Testez et commentez ses performances. Proposez sans le coder un mécanisme que vous jugeriez plus performant.

Question 4 : Création d'images résumés de plan

Exploiter l'ensemble des méthodes vues dans cet Exercice 4 pour mettre en place un mécanisme de génération automatique de résumés de plans dans les vidéos, alternatif au mécanisme de l'Exercice 3. Testez et commentez ses performances. Proposez sans le coder un mécanisme que vous jugeriez plus performant.