TP4

TP 4 - Traitements de vidéos

1 PARTIE 1 : FONCTIONS SIMPLES ET RAISONNEMENT + PRISES EN MAIN D’ OPENCV

1.1 RÉCUPÉRATION DES FRAMES DE LA VIDÉO ET AFFICHAGE

```
1 filename = sys.argv[1] if len(sys.argv) > 1 else '../video.avi'
2 cap = cv.VideoCapture(filename)
```

Il est possible de lire d’ autres vidéo en joignant l’ argument de commande: other video path.

Question 1

run = cap.isOpened()

Making sure the capture has opened successfully

---

if not run:

    # capture opening has failed we cannot do anything :'(

---

print("capture opening has failed we cannot do anything :'(")

---

    sys.exit()

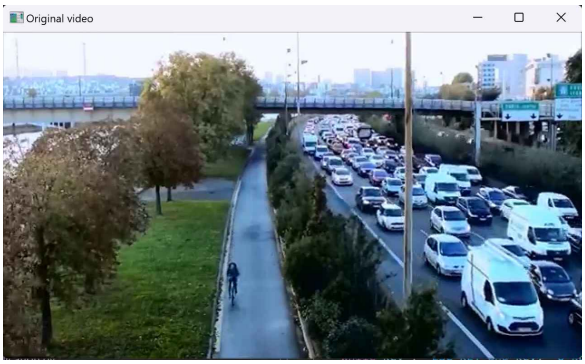
.....

---

    ret, im = cap.read()

```
cv.imshow("Original video", im)
```

Présentation réussie de la vidéo autoroute2.avi:



1.2 EXTRACTION DES INFORMATIONS DE LA VIDÉO

Pour video.avi.

Question 2

```
total_frame= cap.get(cv.CAP_PROP_FRAME_COUNT)
```

Le nombre d’ images qui compose la séquence vidéo N = 614

Question 3

le nombre d’ images qui compose la séquence vidéo

```
total_frame= cap.get(cv.CAP_PROP_FRAME_COUNT)
```

```
# Frame rate de la vidéo.
```

```
fps= cap.get(cv.CAP_PROP_FPS)
```

```
# La durée d’ une vidéo est le nombre total d’ images divisé par le taux d’ images en secondes.
```

```
total_time= total_frame/fps
```

La durée de la séquence vidéo = 20.466666666666665 s

Question 4

```
# Get video wide and high (resolution)

  video_width= cap.get(cv.CAP_PROP_FRAME_WIDTH)

  video_height= cap.get(cv.CAP_PROP_FRAME_HEIGHT)
```

La résolution de la séquence vidéo: 384 x 288

1.3 CONVERTIR LA SÉQUENCE VIDÉO EN NIVEAU DE GRIS

Question 5

```
    ret, im = cap.read()

    # Turning im into grayscale and storing it in imGray

    imGray = cv.cvtColor(im, cv.COLOR_BGR2GRAY)

    cv.imshow("Original video", im)

    cv.imshow("Gray video", imGray)
```

1.4 ENREGISTREMENT DES FRAMES DE LA VIDÉO DANS UN VECTEUR

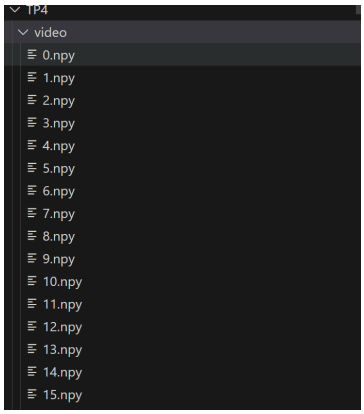
Question 6

```
    index = 0

  # Waiting for the user to press ESCAPE before exiting the application
```

```
while key != ESC_KEY and key!= Q_KEY:
    ret, im = cap.read()
    save_path = os.path.join(save_folder, str(index) + ".npy")
    np.save(save_path, im)
    index += 1
```

Nous avons mis le numpy.array de chaque image dans un fichier npy.




```
frames = []
while key != ESC_KEY and key!= Q_KEY:
    ret, im = cap.read()
    frames.append(im)
```

Enregistre chaque image dans une liste qui peut être lue à partir de la liste si nécessaire. La fonction cap.read() n’ est plus nécessaire.

1.5 EXTRACTION DU FOND DE LA VIDÉO ET DE LA ROUTE

Question 7

```
def moyenne_images(images, M):
```

```
    # Initialiser une variable pour la somme des images
    somme_images = np.zeros_like(images[0], dtype=np.float32)

    # Somme des M premières images
    for i in range(M):
        img = images[i].astype(np.float32)  # Convertir l'image en float32 pour éviter un débordement
        somme_images += img

    # Calculer la moyenne en divisant la somme par le nombre d'images
    moyenne = somme_images / M

    # Convertir en type uint8 si nécessaire
    moyenne_uint8 = np.array(moyenne, dtype=np.uint8)

    return moyenne_uint8
```

Par le code:

M = 200

---

moyenne_img = moyenne_images(gray_frames, M)

On a obtient:

Question 8

Une valeur trop petite de M peut donner une moyenne avec beaucoup de bruit, tandis qu'une valeur trop grande peut lisser excessivement l'image et entraîner une perte de détails.

Question 9