UVSQ ISTY Institut des Sciences et Techniques des Yvelines université PARIS-SACLAY CAMPUS DE MANTES EN YVELINES CAMPUS DE SAINT-QUENTIN-EN-YVELINES	***MACHINE LEARNING***
	Détection faciale

Au cours de ce TP/projet, nous allons utiliser OpenCV (Open Computer Vision) qui est une bibliothèque libre, spécialisée dans le traitement d'images en temps réel.
Dans ce module de Machine Learning ou en français, apprentissage automatique, je tiens à faire une approche pragmatique de cet enseignement. C'est pour cela que nous procédons par étapes :

- étape 1 : les suites récurrentes
- étape 2 : la détection faciale
- étape 3 : la reconnaissance faciale

Les prérequis au TP :
Vous devez avoir Python 3
Vous devez avoir OpenCV 4
Sur votre VM, pour tester :

```
#import de la librairie OpenCV
import cv2
#affichage des informations
print (cv2.getBuildInformation())
```

S'il faut installer OpenCV : `sudo apt-get install python3-opencv` ou `pip install opencv-python`

Comment tester ce programme ?
Évidemment, il existe plusieurs possibilités :

1. avec le terminal linux :
   → dans le mode terminal, saisissez python
   → `>>> import cv2`
   → `>>> print (cv2.getBuildInformation())`
2. avec l'idle de Python
   → ouvrez l'idle
   → file new file
   → enregistrez les 2 lignes dans un programme
   → exécutez le programme par « run » → « run module »
   Note : pour ma part, cette méthode ne fonctionne pas (mais j'en ai l'habitude ;-))
3. avec l'idle de Python et le terminal
   → ouvrez l'idle
   → file new file
   → enregistrez les 2 lignes dans un programme
   → dans le terminal, déplacez vous vers votre répertoire de travail avec `cd` (change directory) et `ls` (**l**ist file**s** : qui affiche le contenu d'un répertoire)
   → une fois arrivé dans votre répertoire, saisissez `python monfichier.py` ce qui va l'exécuter

4. Avec Anaconda :
   → téléchargez la version linux d'anaconda du site https://www.anaconda.com/download
   → cd *Téléchargement*
   → *ls et vous obtenez : Anaconda….Linux-x86_64.sh (un fichier sh est un programme bash)*
   → *pour l'exécuter, bash votrefichier.sh*
   → *puis vous lisez et répondez à toutes les questions*
   → *une fois installé, notez le répertoire où est installé Anaconda*
   → puis *source* ~/ .bashrc
   → pour lancer → conda
   → pour voir les paquets python → conda search « ^python$ » et vous aurez toutes les versions de python listées
   → Ensuite, il peut être intéressant d'installer le notebook….

Exercice 1 :

Testez ce programme. Lorsqu'il fonctionne, commentez toutes les lignes pour comprendre le rôle de chaque instruction de Python et d'OpenCV

```
import cv2
webcam = cv2.VideoCapture(0)
if webcam.isOpened():
  while True:
    bImgReady, imageframe = webcam.read() # get frame per frame from the webcam
    if bImgReady:
      cv2.imshow('My webcam', imageframe) # show the frame
    else:
      print('No image available')
    keystroke = cv2.waitKey(20) # Wait for Key press
    if (keystroke == 27):
      break # if key pressed is ESC then escape the loop

  webcam.release()
  cv2.destroyAllWindows()
```

Exercice 2 :

Voici un exemple du calcul du taux de rafraîchissement de votre caméra :

```
import cv2
#webcam = cv2.VideoCapture(0)
from time import perf_counter
t1_start = perf_counter()
frame_count = 0
webcam = cv2.VideoCapture(0)
NB_IMAGES = 100

if webcam.isOpened():
  while (frame_count < NB_IMAGES):
    bImgReady, imageframe = webcam.read() # get frame per frame from the webcam
    frame_count += 1
```

```
t1_stop = perf_counter()
print ("Frame per Sec.: ", NB_IMAGES / (t1_stop - t1_start))

webcam.release()
cv2.destroyAllWindows()
```

Testez ce programme. Lorsqu'il fonctionne, commentez toutes les lignes pour comprendre le rôle de chaque instruction de Python et d'OpenCV. Normalement, vous devriez obtenir un taux de rafraîchissement juste au-dessus de 25 images par seconde

---

Détection des visages avec OpenCV et HaarCascade

La détection de visage peut être définie comme la capacité à déterminer la position et la taille des visages dans des images numériques, est généralement la première étape clé lors de la création d'applications de traitement de visage (par exemple, la reconnaissance des expressions faciales, la détection de somnolence, la classification de genre, la reconnaissance faciale, l'estimation de la pose de la tête ou l'interaction homme-machine). Cela est dû au fait que ces applications nécessitent en entrée la localisation et la taille des visages détectés. Par conséquent, la détection automatique de visage joue un rôle crucial et constitue l'un des sujets les plus étudiés dans la communauté de l'intelligence artificielle.

La détection de visage semble être une tâche facile pour un être humain, mais c'est une tâche très complexe pour les ordinateurs, car de nombreux problèmes sont généralement impliqués (notamment les variations d'apparence, d'échelle, de rotation, d'expressions faciales, d'occlusion ou encore de conditions d'éclairage). La détection de visage a réalisé des progrès impressionnants après les travaux proposés par **Viola et Jones**, et c'est un indispensable à connaître.

OpenCV propose deux approches pour détecter les visages :
- les détecteurs de visage basés sur les **cascades de Haar (Haar Cascade)**
- les détecteurs de visage basés sur **l'apprentissage en profondeur**(**deep learning**).

Le framework développé par Viola et Jones constitue une méthode efficace pour détecter les objets. Ce framework est très populaire, car **OpenCV** offre des algorithmes de détection de visage basés sur cette méthode. Il peut également être entraîné et utilisé pour détecter d'autres objets que des visages, comme des corps entiers, des plaques d'immatriculation, des parties supérieures du corps ou même des visages de chats. Dans cette section, nous allons voir comment détecter les visages en utilisant ce framework. Nous allons réaliser la détection de visage en utilisant des classificateurs en cascade basés sur les caractéristiques de Haar.

À cet effet, **OpenCV** met à disposition quatre classificateurs en cascade pour la détection de visage (frontale):

1. `haarcascade_frontalface_default.xml`
2. `haarcascade_frontalface_alt.xml`
3. `haarcascade_frontalface_alt2.xml`
4. `haarcascade_frontalface_alt_tree.xml`

Vous pourrez télécharger ces modèles ici:
https://github.com/opencv/opencv/tree/master/data/haarcascades

Dans le prochain exercice, vous allez utiliser le `haarcascade_frontalface_default.xml` . Il suffit donc de le télécharger et de le mettre dans le même répertoire que le répertoire où est votre programme python

Exercice 3 :

Créez une image « test.png » dans le même répertoire. Regardez ce qui se produit

```
import cv2
face_cascade =
cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

#lecture de l'image en entrée
img = cv2.imread('test.png')


gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray, 1.1 , 4)

for (x, y, w , h) in faces:
  cv2.rectangle(img, (x,y),(x+w, y+h), (255, 0, 0) ,3)

#affichage de l'image
cv2.imshow('img', img)
cv2.waitKey()
```

Lorsque le programme fonctionne, commentez toutes les lignes pour comprendre le rôle de chaque instruction de Python et d'OpenCV.

Note :

Vous avez remarqué que l'on pré-traite l'image en la mettant en noir et blanc. Pourquoi?

Réduction de la complexité : La conversion de l'image en noir et blanc réduit la complexité de l'image en supprimant les informations de couleur. Cela permet de simplifier le traitement ultérieur de l'image et de réduire les calculs nécessaires pour détecter les caractéristiques du visage.

Amélioration du contraste : La conversion en noir et blanc peut améliorer le contraste de l'image, ce qui facilite la détection des contours et des caractéristiques du visage. Les variations de couleurs peuvent parfois rendre la détection des motifs plus difficile, tandis que le noir et blanc permet de mettre davantage l'accent sur les différences de luminosité.

Standardisation des données : les modèles Haarcascade sont généralement entraînés sur des images en noir et blanc, donc ils ne seront performants que sur des images similaires.

Petite note: La conversion en noir et blanc n'est pas une étape obligatoire dans tous les cas de **reconnaissance faciale.** Elle dépend du modèle utilisé et des spécificités de la tâche de reconnaissance. Dans certains cas, il peut être préférable de conserver les informations de couleur pour des analyses plus avancées ou des applications spécifiques. Mais dans ces cas là on utilisera généralement **des réseaux de neurones (deep learning).**

Exercice 4 :

Cette fois-ci, ce programme n'utilise pas une image statique mais une vidéo test.mp4. Notez qu'une vidéo n'est qu'une suite d'images statiques. Créez une vidéo« test.mp4 » dans le même répertoire. Regardez ce qui se produit

```
import cv2
face_cascade =
cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

#lecture de l'image en entrée
#img = cv2.imread('test.png')

cap = cv2.VideoCapture('test.mp4')

while cap.isOpened():
  _, img = cap.read()

  gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
  faces = face_cascade.detectMultiScale(gray, 1.1 , 4)

  for (x, y, w , h) in faces:
    cv2.rectangle(img, (x,y),(x+w, y+h), (255, 0, 0) ,3)

  #affichage de l'image
  cv2.imshow('img', img)

  if cv2.waitKey(1) & 0xFF == ord('q'):
    break

cap.release()
```

Lorsque le programme fonctionne, commentez toutes les lignes pour comprendre le rôle de chaque instruction de Python et d'OpenCV

Exercice 5 :

5.1 : à partir du programme de l'exercice 4, avec une image fixe, je vous propose de modifier le programme afin de détecter le contour du visage mais en plus, le contour des yeux

5.2 : à partir du programme de l'exercice 4, avec une image fixe, je vous propose de modifier le programme afin de détecter le contour du visage, le contour des yeux mais aussi le contour du sourire

5.3 : à partir du programme de l'exercice 5, avec une vidéo mp4, je vous propose de modifier le programme afin de détecter le contour du visage mais en plus, le contour des yeux

5.4 : à partir du programme de l'exercice 5, avec une vidéo mp4,, je vous propose de modifier le programme afin de détecter le contour du visage, le contour des yeux mais aussi le contour du sourire

5.4 : Vous est-il possible de faire la même chose mais avec votre webcam ????