

1. Introduction

[OpenCV](#) (pour Open Computer Vision) est une bibliothèque graphique libre, initialement développée par Intel, spécialisée dans le traitement d'images en temps réel.

Voici une liste non exhaustive des différentes opérations d'images que permet de faire OpenCV :

- Ouverture, enregistrement et affichage d'une image ou d'une vidéo.
- Construction et affichage de l'histogramme d'une image
- Localisation et extraction d'objets.
- Opérations d'image usuelles (redimensionnement, rotation, opérateurs morphologiques, filtrage, etc).
- Détection et suivi (tracking) d'objets
- Algorithmes usuels de [machine learning](#) tels que les [réseaux de neurones](#).

Comparé à d'autres langages comme C / C ++, Python est 10 fois plus lent. Mais une autre caractéristique importante de Python est qu'il est interfaçable avec C / C ++. ([wrapper](#) Python). Cela donne deux avantages :

- Le code est aussi rapide que le code C / C ++ original (puisque c'est le code C ++ qui fonctionne en arrière-plan)
- Il est très facile de coder en Python. C'est ainsi que fonctionne la bibliothèque [OpenCV-Python](#), c'est un [wrapper](#) Python autour de l'implémentation C ++ originale.

OpenCV n'est pas un magicien, si les images sont mauvaises, le résultat le sera aussi. Traiter des images en 1600x1200 donnera un superbe résultat, mais au prix d'une surcharge de calcul importante. Si les mathématiques derrière vos algorithmes sont incorrectes, le résultat le sera aussi. Utiliser des algorithmes pré implémentés est intéressant, mais il faut tout de même comprendre ce qu'ils font.

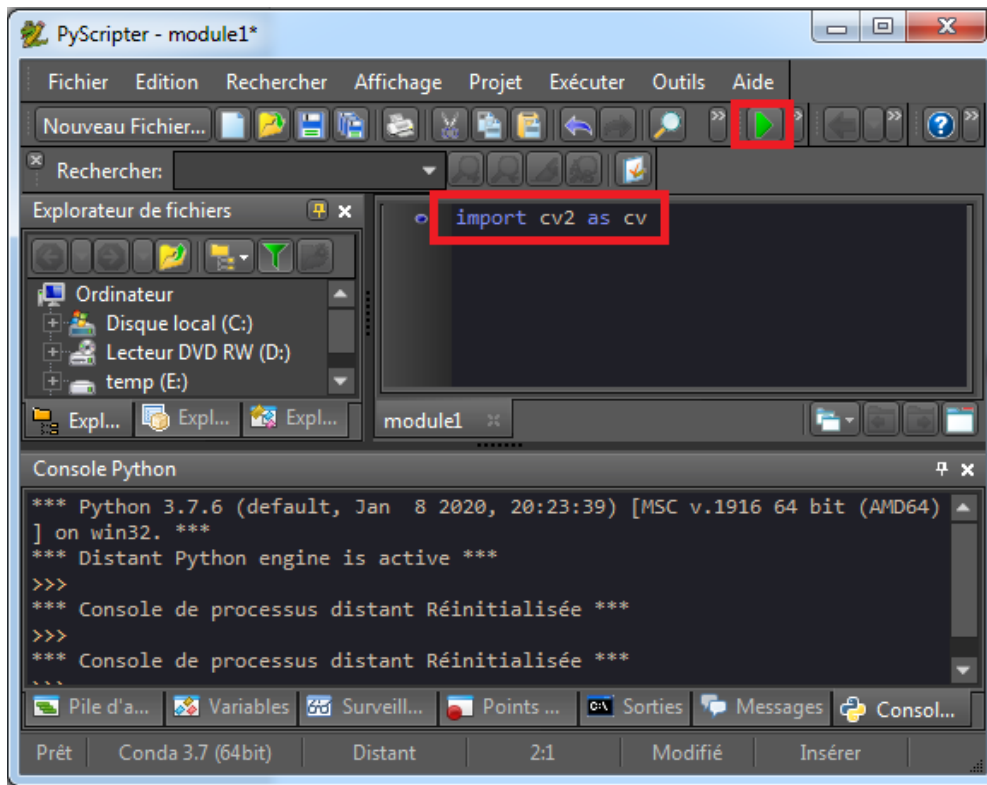
Ce TP a été écrit à l'aide de la [vidéo suivante](#) et des exemples disponibles sur [GitHub](#).



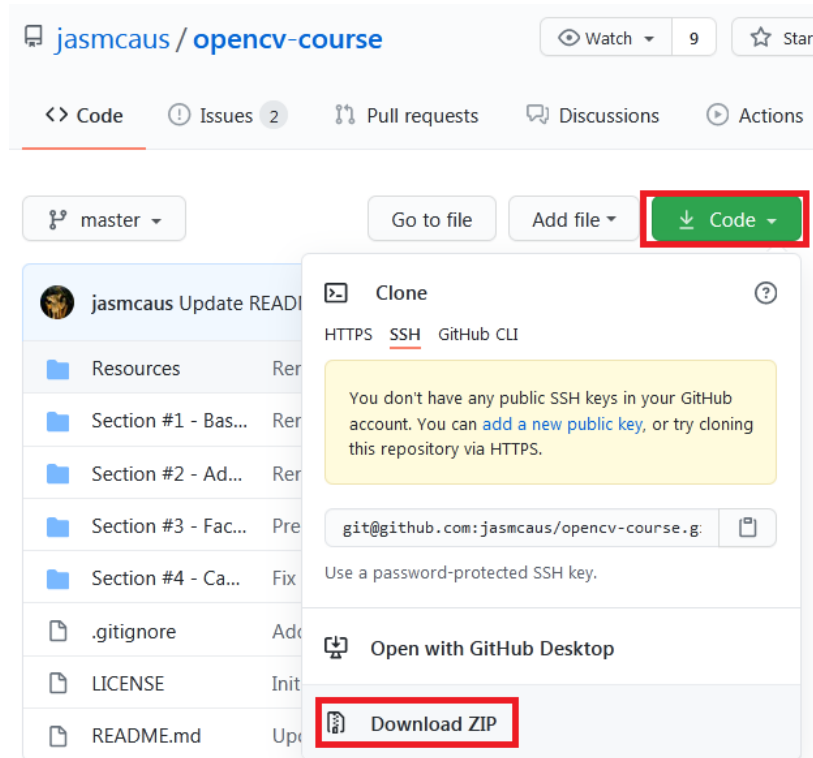
2.Prérequis

2.1 Vérifier le bon fonctionnement de la bibliothèque OpenCV avec Edupython3.

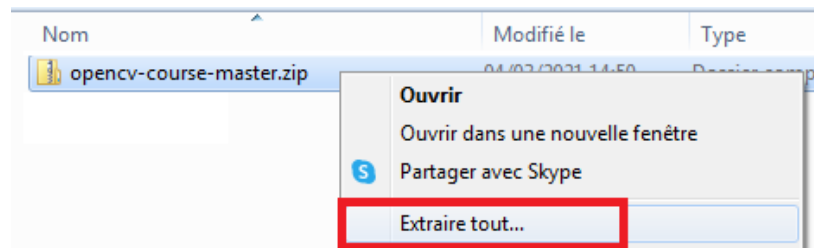
Quand on exécute le script, il ne doit pas y avoir de messages d'erreur.



2.2 Télécharger les exemples sur le site [GitHub](#) de Jasmcaus.



2.3 Déplacer le fichier téléchargé dans Documents\NSI puis décompresser le fichier ZIP



| | |
|-----------------------|--|
| essais | Répertoire essais à créer qui contiendra vos programmes |
| Resources | Fichiers images et vidéo |
| Section #1 - Basics | Corrigés de la vidéo YouTube |
| Section #2 - Advanced | |
| Section #3 - Faces | |
| Section #4 - Capstone | |

2.4 Programme afficher une image

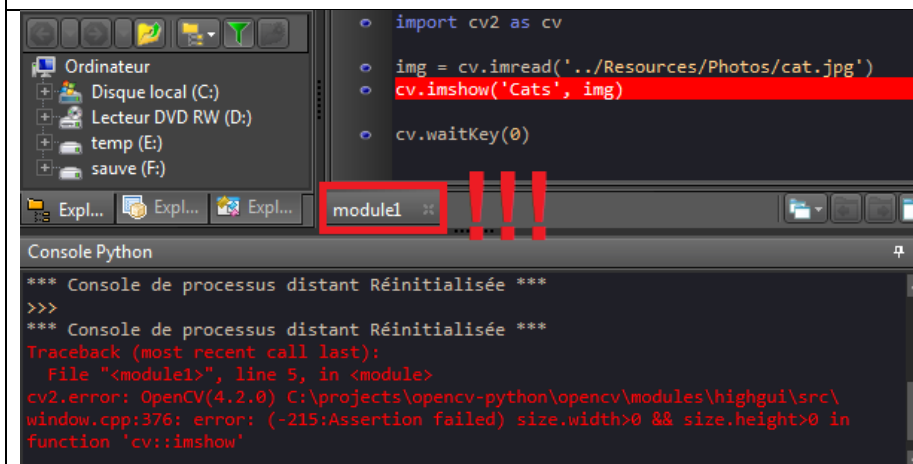
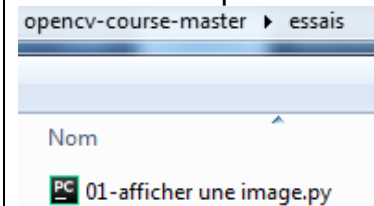
| Nom du programme : 01-afficher une image.py | Résultat affiché |
|---|------------------|
| <pre>#Afficher une image import cv2 as cv img = cv.imread('../Resources/Photos/cat.jpg') cv.imshow('Cats', img) cv.waitKey(0) cv.destroyAllWindows()</pre> | |

Le répertoire essai contient le fichier python

01-afficher une image.py

Le fichier python doit être sauvegardé **AVANT** exécution

Contenu du répertoire essais



Ici, le programme n'est pas sauvegardé, le nom est module1, le chemin pour accéder à l'image ne peut pas être bon.

Error

3.Exemples Basics

3.1 Afficher une vidéo


Touche d minuscule pour stopper la vidéo.

Dans la vidéo YouTube, une erreur d'exécution apparait, c'est normal. La version ci-dessous corrige cette erreur. En cas de problème, appuyer sur Ctrl +F2 pour réinitialiser Edupython3.


| Nom du programme : 02-afficher une video.py | Résultat affiché |
|--|---|
| <pre>#Afficher une video import cv2 as cv capture = cv.VideoCapture('../Resources/Videos/dog.mp4') while capture.isOpened(): ret, frame = capture.read() if ret==True: cv.imshow('Video', frame) if cv.waitKey(20) & 0xFF==ord('d'): break else: break capture.release() cv.destroyAllWindows()</pre> |  |

3.2 Afficher une vidéo en noir et blanc

Tester le code suivant.

| Nom du programme : 03-afficher une video noir et blanc.py | Résultat affiché |
|--|---|
| <pre>#Afficher une video en noir et blanc import cv2 as cv capture = cv.VideoCapture('../Resources/Videos/dog.mp4') while capture.isOpened(): ret, frame = capture.read() if ret==True: gray = cv.cvtColor(frame, cv.COLOR_BGR2GRAY) cv.imshow('Video', gray) if cv.waitKey(20) & 0xFF==ord('d'): break else: break capture.release() cv.destroyAllWindows()</pre> |  |

3.3 Redimensionner une image

| Nom du programme : 04-Redimensionner une image.py | Résultat affiché |
|--|---|
| <pre>#Redimensionner une image import cv2 as cv img = cv.imread('../Resources/Photos/cat.jpg') def rescaleFrame(frame, scale=0.75): width = int(frame.shape[1]*scale) height = int(frame.shape[0]*scale) dimensions = (width,height) return cv.resize(frame,dimensions,interpolation=cv.INTER_AREA) img_resized=rescaleFrame(img) cv.imshow('Cat1', img) cv.imshow('Cat2', img_resized) cv.waitKey(0) cv.destroyAllWindows()</pre> |  |

Redimensionner l'image de la fenêtre « cat2 » à 20%

3.4 Redimensionner une vidéo

A l'aide de la vidéo Youtube ([0:12:57](#)) Resizing and Rescaling Frames, compléter le programme ci-dessous en redimensionnant l'affichage de la vidéo à 20%.

| Nom du programme : 05-Redimensionner une video.py | Résultat affiché |
|---|---|
| <pre>#Redimensionner une video import cv2 as cv capture = cv.VideoCapture('../Resources/Videos/dog.mp4') def rescaleFrame(frame, scale=0.75): width = int(frame.shape[1]*scale) height = int(frame.shape[0]*scale) dimensions = (width,height) return cv.resize(frame,dimensions,interpolation=cv.INTER_AREA) while capture.isOpened(): ret, frame = capture.read() if ret==True: #A compléter if cv.waitKey(20) & 0xFF==ord('d'): break else: break capture.release() cv.destroyAllWindows()</pre> |  |

3.5 Dessiner des formes géométriquesNom du programme : **06-Dessiner.py**

```
#Dessiner
import cv2 as cv
import numpy as np

blank = np.zeros((500,500,3), dtype='uint8')

# 1. Paint the image a certain colour
blank[:] = 128,128,128

# 2. Draw a Rectangle
cv.rectangle(blank, (0,0), (blank.shape[1]//3, blank.shape[0]//3), (0,255,0), thickness=-1)

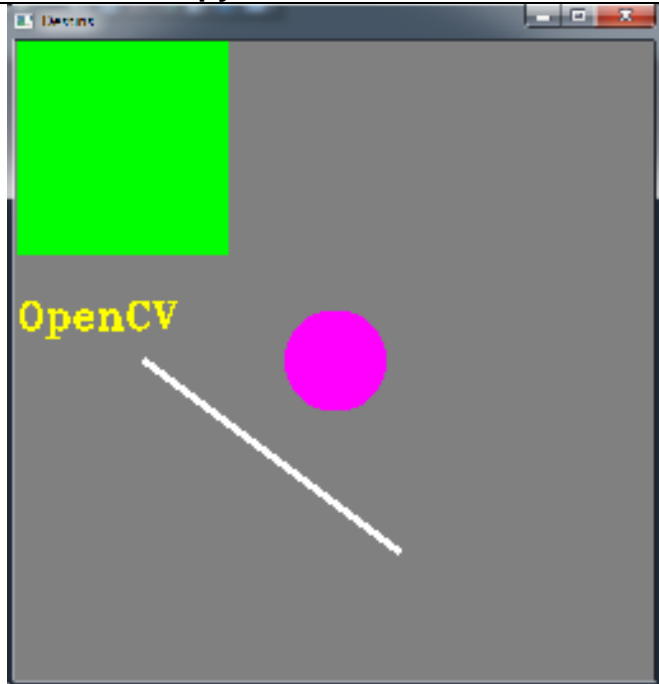
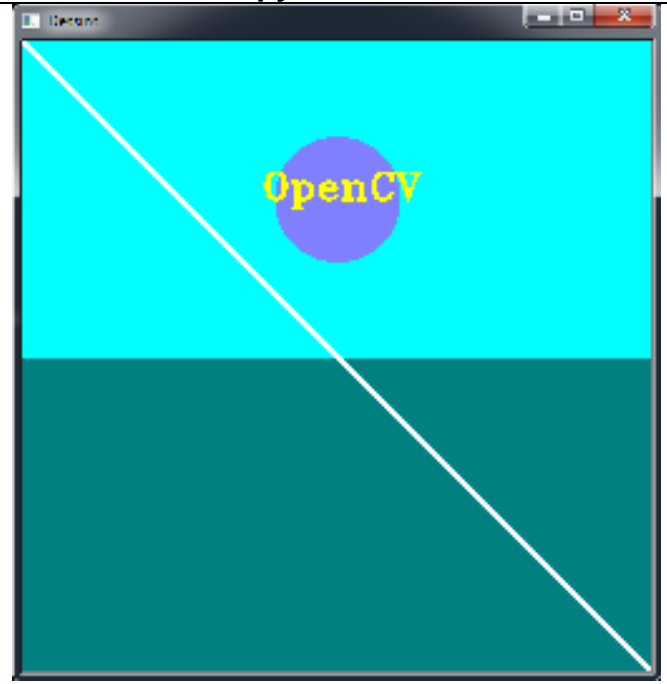
# 3. Draw A circle
cv.circle(blank, (blank.shape[1]//2, blank.shape[0]//2), 40, (255,0,255), thickness=-1)

# 4. Draw a line
cv.line(blank, (100,250), (300,400), (255,255,255), thickness=3)

# 5. Write text
cv.putText(blank, 'OpenCV', (0,225), cv.FONT_HERSHEY_TRIPLEX, 1.0, (0,255,255), 2)
cv.imshow('Dessins', blank)

cv.waitKey(0)
cv.destroyAllWindows()
```

Modifier le programme précédent afin d'afficher l'image de droite

Résultat affiché
06-Dessiner.py

Résultat affiché
07-Dessiner_v2.py


3.6 Les 5 fonctions essentielles d'OpenCV

| Function | Fonction |
|-------------------------|-------------------------------|
| Converting to grayscale | Conversion en niveaux de gris |
| Blur | Flou |
| Edge Cascade | Détourage |
| Dilating the image | Dilatation |
| Eroding | Erosion |
| Resize | Redimensionner |
| Cropping | Recadrage |

Nom du programme : 08-basic_functions.py

```
import cv2 as cv

img = cv.imread('../Resources/Photos/park.jpg')
cv.imshow('Park', img)

# Converting to grayscale
gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
cv.imshow('Gray', gray)

# Blur
blur = cv.GaussianBlur(img, (7,7), cv.BORDER_DEFAULT)
cv.imshow('Blur', blur)

# Edge Cascade
canny = cv.Canny(blur, 125, 175)
cv.imshow('Canny Edges', canny)

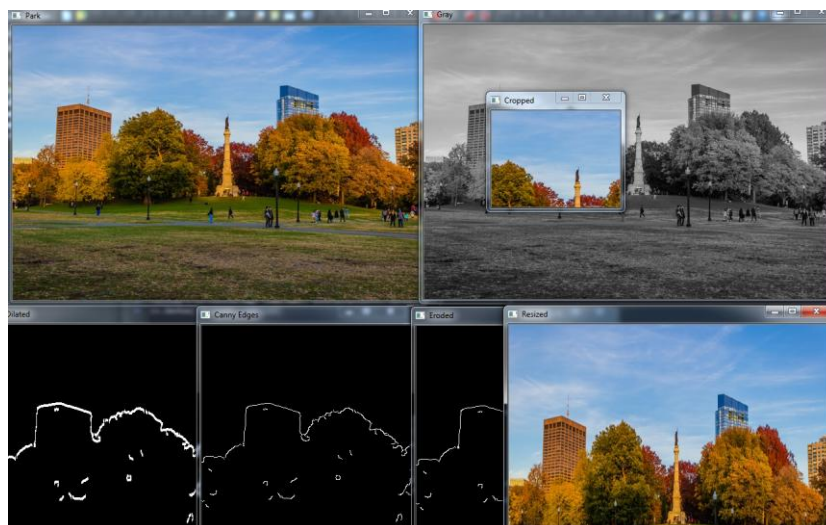
# Dilating the image
dilated = cv.dilate(canny, (7,7), iterations=3)
cv.imshow('Dilated', dilated)

# Eroding
eroded = cv.erode(dilated, (7,7), iterations=3)
cv.imshow('Eroded', eroded)

# Resize
resized = cv.resize(img, (500,500), interpolation=cv.INTER_CUBIC)
cv.imshow('Resized', resized)

# Cropping
cropped = img[50:200, 200:400]
cv.imshow('Cropped', cropped)

cv.waitKey(0)
cv.destroyAllWindows()
```



3.7 transformations des images

Nom du programme : 09-transformations.py

```

import cv2 as cv
import numpy as np
img = cv.imread('../Resources/Photos/park.jpg')
cv.imshow('Park', img)

# Translation
def translate(img, x, y):
    transMat = np.float32([[1,0,x],[0,1,y]])
    dimensions = (img.shape[1], img.shape[0])
    return cv.warpAffine(img, transMat, dimensions)
# -x --> Left # -y --> Up # x --> Right # y --> Down

translated = translate(img, -100, 100)
cv.imshow('Translated', translated)

# Rotation
def rotate(img, angle, rotPoint=None):
    (height,width) = img.shape[:2]

    if rotPoint is None:
        rotPoint = (width//2,height//2)

    rotMat = cv.getRotationMatrix2D(rotPoint, angle, 1.0)
    dimensions = (width,height)

    return cv.warpAffine(img, rotMat, dimensions)

rotated = rotate(img, -45)
cv.imshow('Rotated', rotated)

rotated_rotated = rotate(img, -90)
cv.imshow('Rotated Rotated', rotated_rotated)

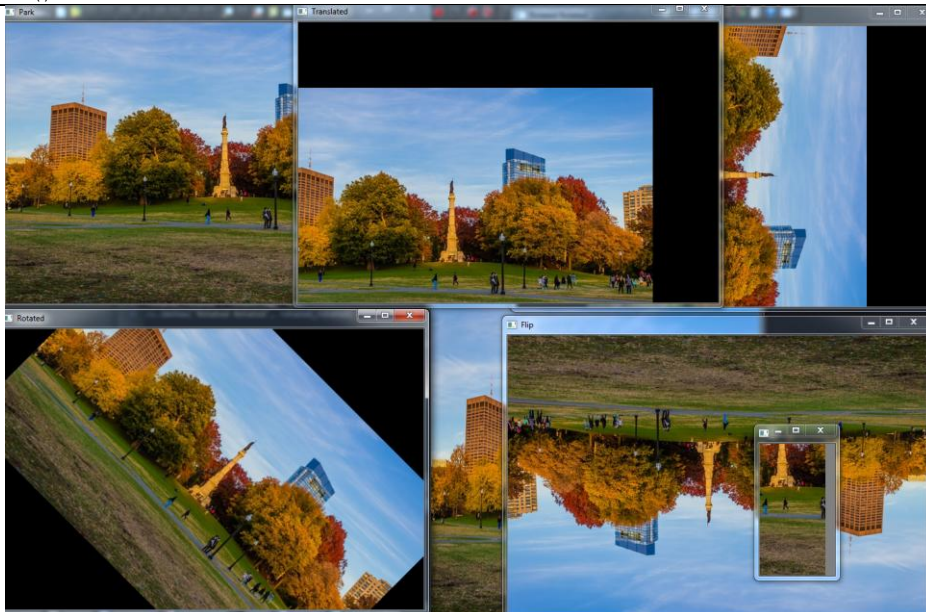
# Resizing
resized = cv.resize(img, (500,500), interpolation=cv.INTER_CUBIC)
cv.imshow('Resized', resized)

# Flipping
flip = cv.flip(img, -1)
cv.imshow('Flip', flip)

# Cropping
cropped = img[200:400, 300:400]
cv.imshow('Cropped', cropped)

cv.waitKey(0)
cv.destroyAllWindows()

```



3.8 Détection de contours

Nom du programme : 10-contours.py

```

import cv2 as cv
import numpy as np

img = cv.imread('woman.jpg')
cv.imshow('Cats', img)

blank = np.zeros(img.shape, dtype='uint8')
cv.imshow('Blank', blank)

gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
cv.imshow('Gray', gray)

blur = cv.GaussianBlur(gray, (5,5), cv.BORDER_DEFAULT)
cv.imshow('Blur', blur)

canny = cv.Canny(blur, 125, 175)
cv.imshow('Canny Edges', canny)

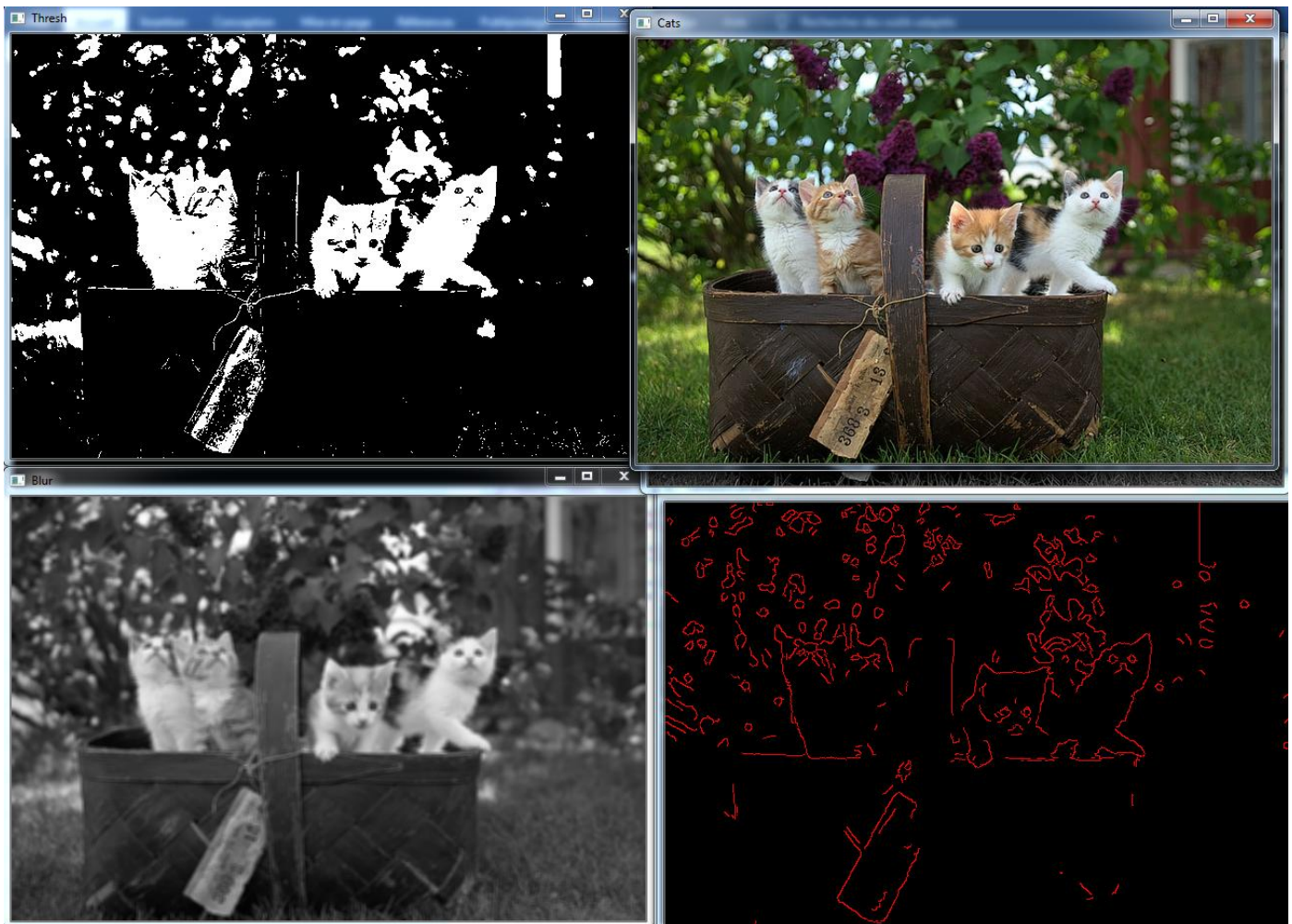
ret, thresh = cv.threshold(gray, 125, 255, cv.THRESH_BINARY)
cv.imshow('Thresh', thresh)

contours, hierarchies = cv.findContours(canny, cv.RETR_LIST, cv.CHAIN_APPROX_SIMPLE)
print(f'{len(contours)} contour(s) found!')


cv.drawContours(blank, contours, -1, (0,0,255), 1)
cv.imshow('Contours Drawn', blank)

cv.waitKey(0)
cv.destroyAllWindows()

```

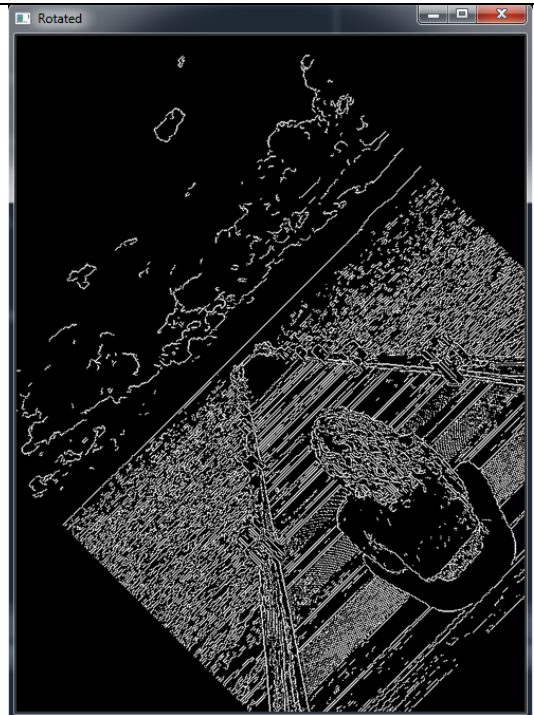


3.9 Exercice

A partir d'une image [libre de droit](#) de votre choix ( 480×640), effectuer une extraction de contours avec une rotation de +45 degrés.

Nom du programme : **11-contours_ex.py**

#Coller votre programme ici



4. Exemples avancés

4.1 traitement en [fausses couleurs](#)

| Function | Fonction |
|----------------------------|--|
| Grayscale | Niveaux de gris |
| HSV (Hue Saturation Value) | Teinte Saturation Valeur |
| LAB | CIELAB |
| RGB | Rouge vert bleu |
| BGR | Bleu vert rouge |

Nom du programme : 12-colour_spaces.py

```
import cv2 as cv
import matplotlib.pyplot as plt

img = cv.imread('../Resources/Photos/park.jpg')
cv.imshow('Park', img)

# plt.imshow(img)
# plt.show()

# BGR to Grayscale
gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
cv.imshow('Gray', gray)

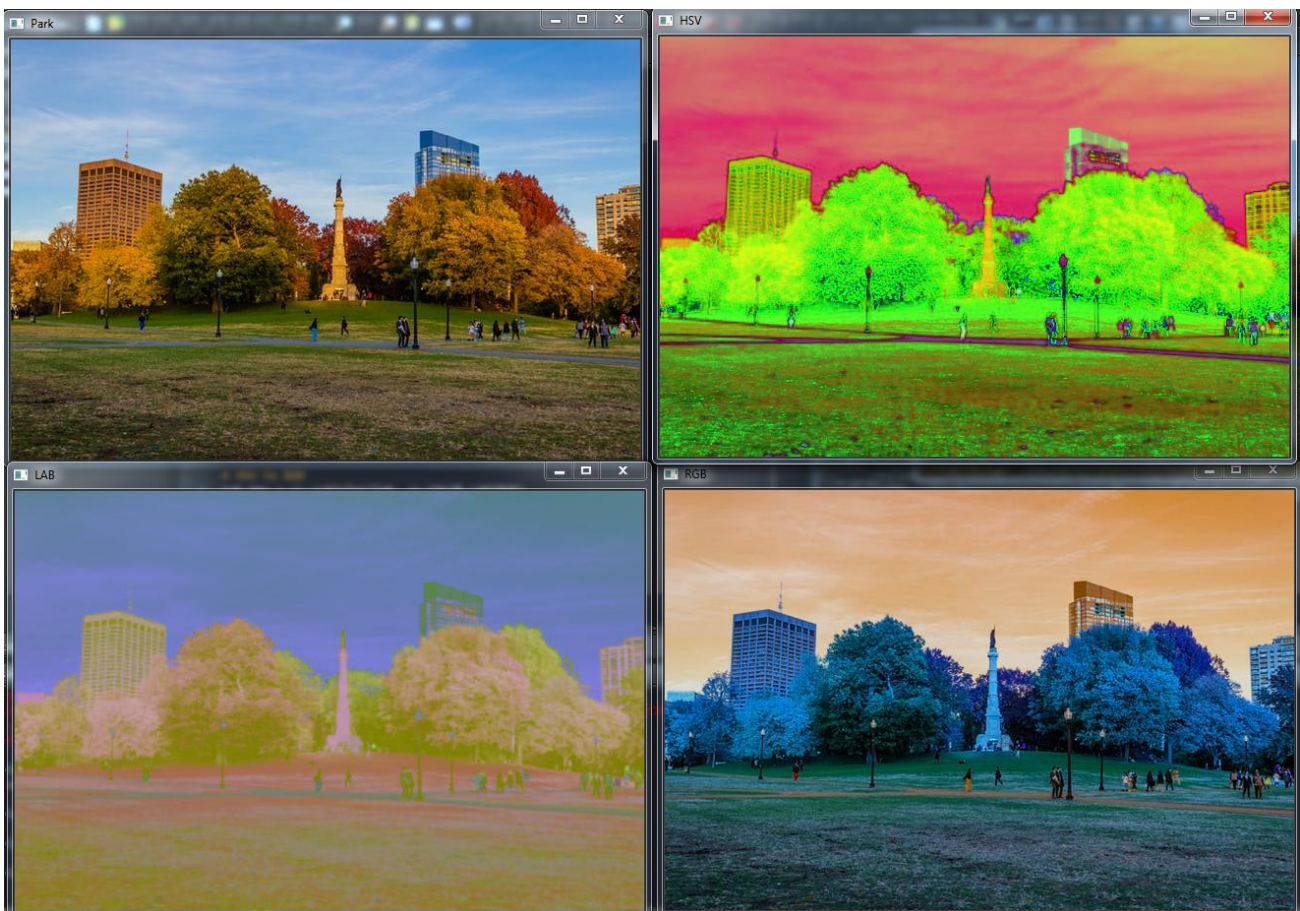
# BGR to HSV
hsv = cv.cvtColor(img, cv.COLOR_BGR2HSV)
cv.imshow('HSV', hsv)

# BGR to L*a*b*
lab = cv.cvtColor(img, cv.COLOR_BGR2LAB)
cv.imshow('LAB', lab)

# BGR to RGB
rgb = cv.cvtColor(img, cv.COLOR_BGR2RGB)
cv.imshow('RGB', rgb)

# HSV to BGR
lab_bgr = cv.cvtColor(lab, cv.COLOR_LAB2BGR)
cv.imshow('LAB --> BGR', lab_bgr)

cv.waitKey(0)
cv.destroyAllWindows()
```



4.2 Séparation BGR et regroupement BGR

Nom du programme : 13-colour_spaces.py

```
import cv2 as cv
import numpy as np

img = cv.imread('../Resources/Photos/park.jpg')
cv.imshow('Park', img)

blank = np.zeros(img.shape[:2], dtype='uint8')

b,g,r = cv.split(img)

blue = cv.merge([b,blank,blank])
green = cv.merge([blank,g,blank])
red = cv.merge([blank,blank,r])

cv.imshow('Blue', blue)
cv.imshow('Green', green)
cv.imshow('Red', red)

print(img.shape)
print(b.shape)
print(g.shape)
print(r.shape)

merged = cv.merge([b,g,r])
cv.imshow('Merged Image', merged)

cv.waitKey(0)
cv.destroyAllWindows()
```

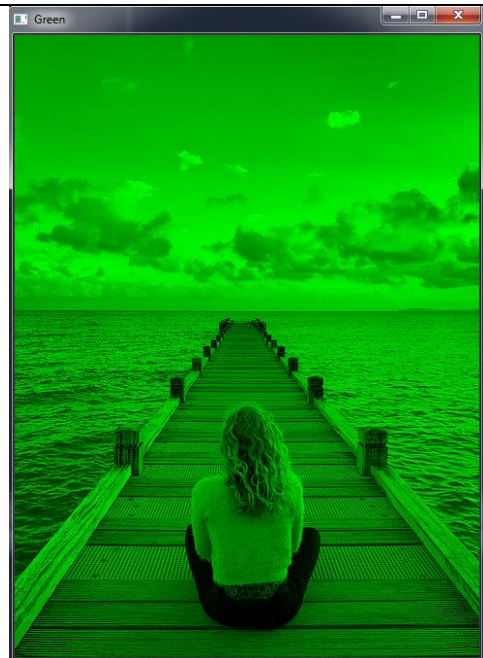


4.3 Exercice

A partir d'une image [libre de droit](#) de votre choix (480×640), effectuer une extraction de la couleur verte.

Nom du programme : 14-colour_spaces_ex.py

#Coller votre programme ici



4.4 Masque flou

Nom du programme : 15-blurring.py

```
import cv2 as cv

img = cv.imread('../Resources/Photos/cats.jpg')
cv.imshow('Cats', img)

# Averaging
average = cv.blur(img, (3,3))
cv.imshow('Average Blur', average)

# Gaussian Blur
gauss = cv.GaussianBlur(img, (3,3), 0)
cv.imshow('Gaussian Blur', gauss)

# Median Blur
median = cv.medianBlur(img, 3)
cv.imshow('Median Blur', median)

# Bilateral
bilateral = cv.bilateralFilter(img, 10, 35, 25)
cv.imshow('Bilateral', bilateral)

cv.waitKey(0)
cv.destroyAllWindows()
```



4.5 Opérations logiques sur des masques

Nom du programme : 16-bitwise.py

```
import cv2 as cv
import numpy as np

blank = np.zeros((400,400), dtype='uint8')

rectangle = cv.rectangle(blank.copy(), (30,30), (370,370),
255, -1)
circle = cv.circle(blank.copy(), (200,200), 200, 255, -1)

cv.imshow('Rectangle', rectangle)
cv.imshow('Circle', circle)

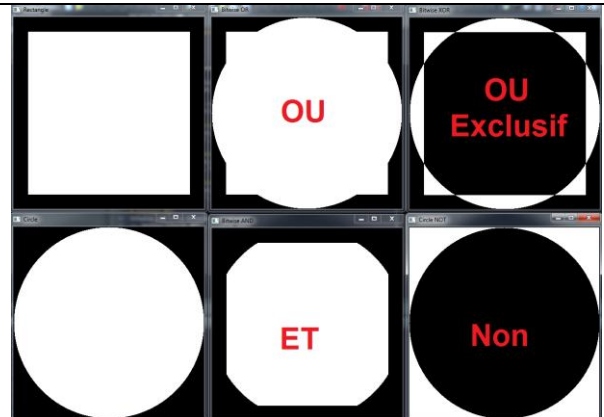
# bitwise AND --> intersecting regions
bitwise_and = cv.bitwise_and(rectangle, circle)
cv.imshow('Bitwise AND', bitwise_and)

# bitwise OR --> non-intersecting and intersecting regions
bitwise_or = cv.bitwise_or(rectangle, circle)
cv.imshow('Bitwise OR', bitwise_or)

# bitwise XOR --> non-intersecting regions
bitwise_xor = cv.bitwise_xor(rectangle, circle)
cv.imshow('Bitwise XOR', bitwise_xor)

# bitwise NOT
bitwise_not = cv.bitwise_not(circle)
cv.imshow('Circle NOT', bitwise_not)

cv.waitKey(0)
cv.destroyAllWindows()
```



4.6 Opérations logiques sur les images

Nom du programme : 17-masking.py

```
import cv2 as cv
import numpy as np

img = cv.imread('../Resources/Photos/cats.jpg')
cv.imshow('Cats', img)

blank = np.zeros(img.shape[:2], dtype='uint8')
cv.imshow('Blank Image', blank)

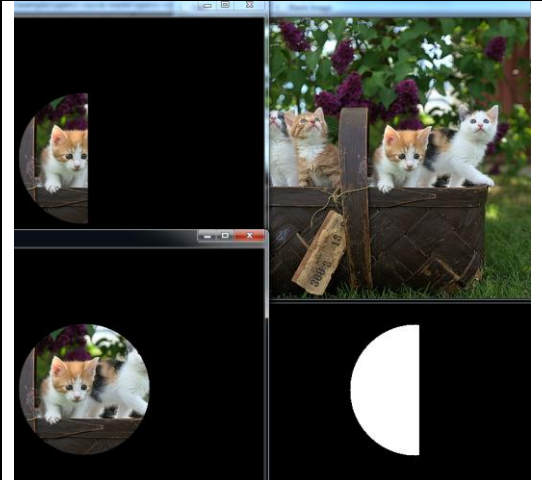
circle = cv.circle(blank.copy(),
                    (img.shape[1]//2+45, img.shape[0]//2), 100, 255, -1)

rectangle = cv.rectangle(blank.copy(), (30,30), (370,370), 255, -1)

weird_shape = cv.bitwise_and(circle, rectangle)
cv.imshow('Weird Shape', weird_shape)

masked = cv.bitwise_and(img, img, mask=weird_shape)
cv.imshow('Weird Shaped Masked Image', masked)

masked = cv.bitwise_and(img, img, mask=circle)
cv.imshow('Weird Shaped Masked Image2', masked)
cv.waitKey(0)
cv.destroyAllWindows()
```



A voir : https://docs.opencv.org/master/d0/d86/tutorial_py_image_arithmetics.html

4.7 Histogrammes sur les images

4.7.1 Histogramme des niveaux de gris

Nom du programme : 18-histogram_gray.py

```
import cv2 as cv
import matplotlib.pyplot as plt
import numpy as np

img = cv.imread('../Resources/Photos/cats.jpg')
cv.imshow('Cats', img)

blank = np.zeros(img.shape[:2], dtype='uint8')

gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
cv.imshow('Gray', gray)

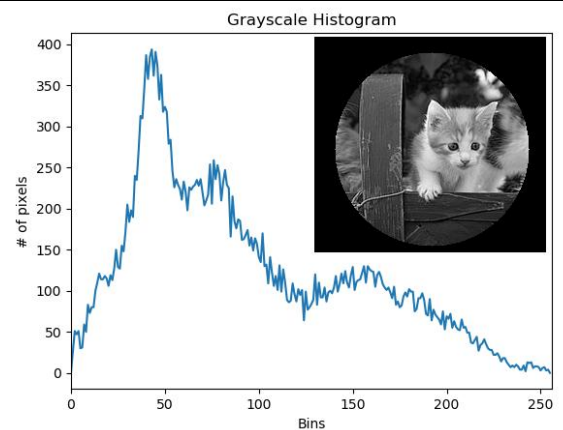
mask = cv.circle(blank, (img.shape[1]//2, img.shape[0]//2),
                  100, 255, -1)

masked = cv.bitwise_and(gray, gray, mask=mask)
cv.imshow('Mask', masked)

# Grayscale histogram
gray_hist = cv.calcHist([gray], [0], mask, [256], [0,256] )

plt.figure()
plt.title('Grayscale Histogram')
plt.xlabel('Bins')
plt.ylabel('# of pixels')
plt.plot(gray_hist)
plt.xlim([0,256])
plt.show()

cv.waitKey(0)
cv.destroyAllWindows()
```



4.7.2 Histogramme des couleurs

Nom du programme : 19-histogram_colour.py

```
import cv2 as cv
import matplotlib.pyplot as plt
import numpy as np

img = cv.imread('../Resources/Photos/cats.jpg')
cv.imshow('Cats', img)

blank = np.zeros(img.shape[:2], dtype='uint8')

mask = cv.circle(blank, (img.shape[1]//2, img.shape[0]//2), 100,
255, -1)

masked = cv.bitwise_and(img, img, mask=mask)
cv.imshow('Mask', masked)

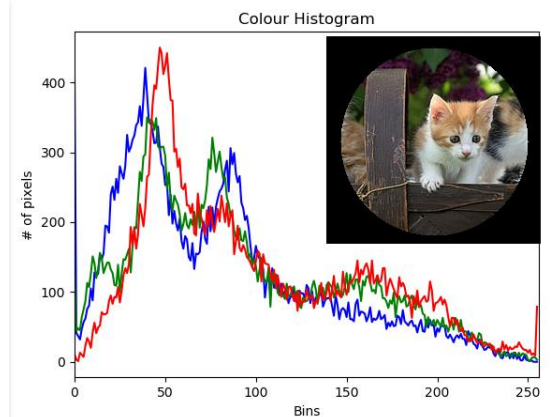
# Colour Histogram

plt.figure()
plt.title('Colour Histogram')
plt.xlabel('Bins')
plt.ylabel('# of pixels')
colors = ('b', 'g', 'r')
for i, col in enumerate(colors):
    hist = cv.calcHist([img], [i], mask, [256], [0, 256])
    plt.plot(hist, color=col)
    plt.xlim([0, 256])

plt.show()

cv.waitKey(0)
cv.destroyAllWindows()

cv.waitKey(0)
```



4.7 Détection de contours

Nom du programme : 20-gradients.py

```
import cv2 as cv
import numpy as np

img = cv.imread('../Resources/Photos/park.jpg')
cv.imshow('Park', img)

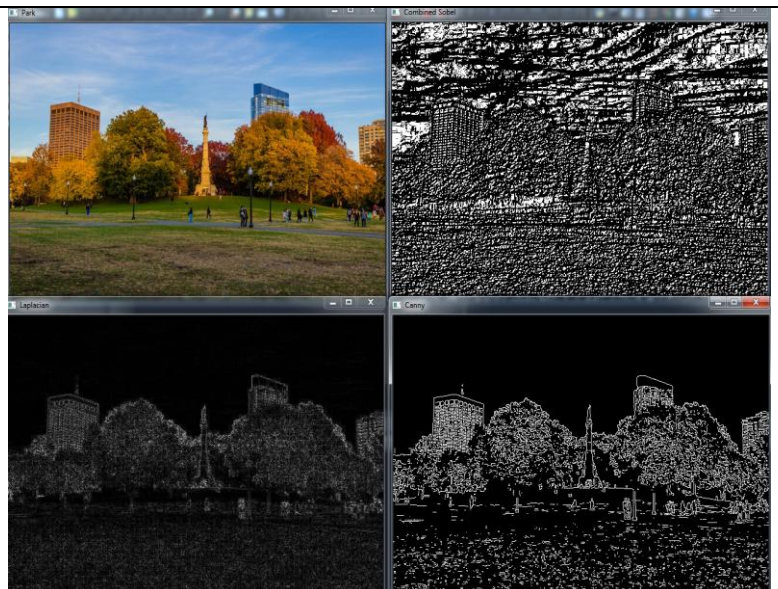
gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
cv.imshow('Gray', gray)

# Laplacian
lap = cv.Laplacian(gray, cv.CV_64F)
lap = np.uint8(np.absolute(lap))
cv.imshow('Laplacian', lap)

# Sobel
sobelx = cv.Sobel(gray, cv.CV_64F, 1, 0)
sobely = cv.Sobel(gray, cv.CV_64F, 0, 1)
combined_sobel = cv.bitwise_or(sobelx, sobely)

cv.imshow('Sobel X', sobelx)
cv.imshow('Sobel Y', sobely)
cv.imshow('Combined Sobel', combined_sobel)

canny = cv.Canny(gray, 150, 175)
cv.imshow('Canny', canny)
cv.waitKey(0)
cv.destroyAllWindows()
```



5. Détection de visages

5.1 Détection de visages dans une image

Attention il faut que le fichier `haar_face.xml` soit dans le même répertoire que le fichier python.

Le fichier `haar_face.xml` est un [classificateur](#) au format xml indiquant la structure de données à rechercher dans une image, ici, des visages.

Il est possible [de créer soi-même des classificateurs](#) pour openCV.

Nom du programme : 21-detectionVisages.py

```
import cv2 as cv

img = cv.imread('../Resources/Photos/group 2.jpg')
cv.imshow('Group of 5 people', img)

gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
cv.imshow('Gray People', gray)

haar_cascade = cv.CascadeClassifier('haar_face.xml')

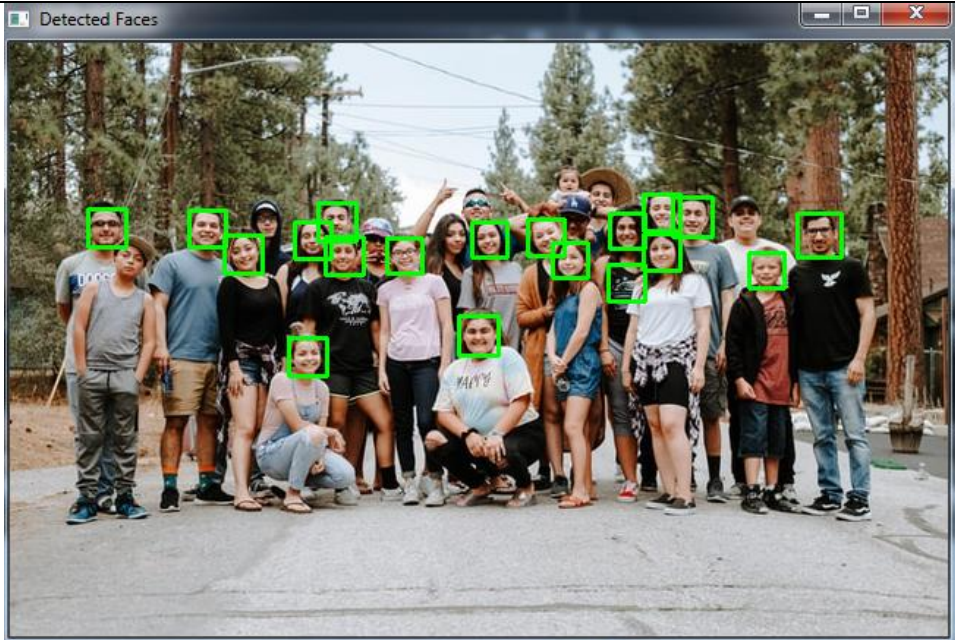
faces_rect = haar_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=1)

print(f'Number of faces found = {len(faces_rect)}')

for (x,y,w,h) in faces_rect:
    cv.rectangle(img, (x,y), (x+w,y+h), (0,255,0), thickness=2)

cv.imshow('Detected Faces', img)

cv.waitKey(0)
cv.destroyAllWindows()
```

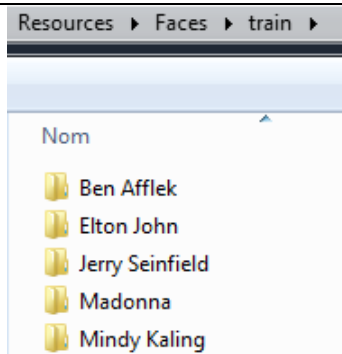


Le logiciel n'a pas reconnu tous les visages. En effet il est nécessaire que l'image soit de bonne résolution.

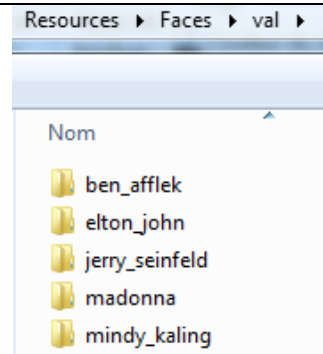
5.2 Exercice : Prenez une photo du groupe NSI afin d'appliquer le programme de détection de visages.

5.3 Reconnaissance de visages dans une image : apprentissage

Le répertoire **train** contient des images de 5 personnes. Ce répertoire est utilisé pour générer un fichier d'apprentissage **yaml**



Le répertoire **val** contient des images des mêmes personnes, mais dans des postures différentes. Ces images vont être utilisées pour vérifier si le logiciel reconnaît bien les visages et par conséquent le nom de la personne en photo.



Nom du programme : **opencv-course-master\Section #3 – Faces\faces_train.py**

```
import os
import cv2 as cv
import numpy as np

people = ['Ben Afflek', 'Elton John', 'Jerry Seinfeld', 'Madonna', 'Mindy Kaling']
DIR = r'../Resources/Faces/train'

haar_cascade = cv.CascadeClassifier('haar_face.xml')

features = []
labels = []

def create_train():
    for person in people:
        path = os.path.join(DIR, person)
        label = people.index(person)

        for img in os.listdir(path):
            img_path = os.path.join(path, img)

            img_array = cv.imread(img_path)
            if img_array is None:
                continue

            gray = cv.cvtColor(img_array, cv.COLOR_BGR2GRAY)

            faces_rect = haar_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=4)

            for (x,y,w,h) in faces_rect:
                faces_roi = gray[y:y+h, x:x+w]
                features.append(faces_roi)
                labels.append(label)

create_train()
print('Training done -----')

features = np.array(features, dtype='object')
labels = np.array(labels)

face_recognizer = cv.face.LBPHFaceRecognizer_create()

# Train the Recognizer on the features list and the labels list
face_recognizer.train(features, labels)

face_recognizer.save('face_trained.yml')
np.save('features.npy', features)
np.save('labels.npy', labels)
```

```
*** Console de processus distant Réinitialisée ***
Training done -----
>>>
```

5.4 Reconnaissance de visages dans une image : vérifications

Nom du programme : **opencv-course-master\Section #3 – Faces\faces_train.py**

```

import numpy as np
import cv2 as cv

haar_cascade = cv.CascadeClassifier('haar_face.xml')

people = ['Ben Afflek', 'Elton John', 'Jerry Seinfeld', 'Madonna', 'Mindy Kaling']
# features = np.load('features.npy', allow_pickle=True)
# labels = np.load('labels.npy')

face_recognizer = cv.face.LBPHFaceRecognizer_create()
face_recognizer.read('face_trained.yml')

#img = cv.imread(r'../Resources\Faces\val\elton_john/1.jpg')
img = cv.imread(r'../Resources\Faces\val\ben_afflek/2.jpg')
#img = cv.imread(r'../Resources\Faces\val\madonna/4.jpg')

gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
cv.imshow('Person', gray)

# Detect the face in the image
faces_rect = haar_cascade.detectMultiScale(gray, 1.1, 4)

for (x,y,w,h) in faces_rect:
    faces_roi = gray[y:y+h,x:x+w]

    label, confidence = face_recognizer.predict(faces_roi)
    print(f'Label = {people[label]} with a confidence of {confidence}')

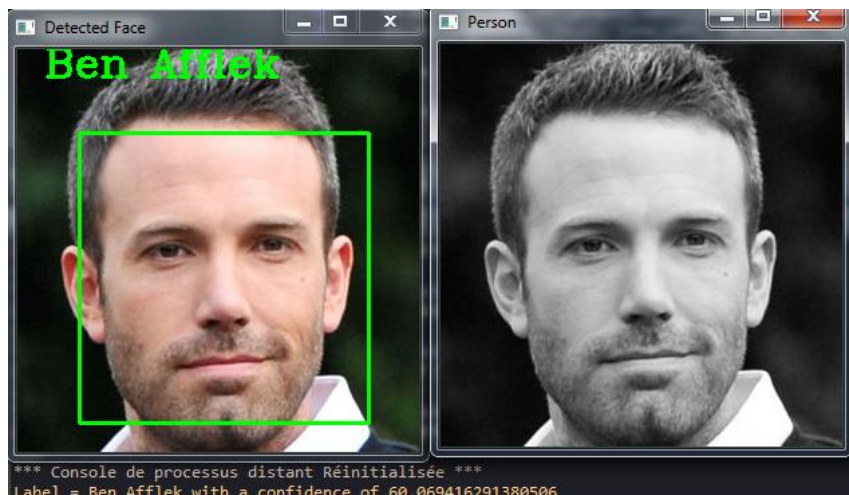
    cv.putText(img, str(people[label]), (20,20), cv.FONT_HERSHEY_COMPLEX, 1.0, (0,255,0), thickness=2)
    cv.rectangle(img, (x,y), (x+w,y+h), (0,255,0), thickness=2)

cv.imshow('Detected Face', img)

cv.waitKey(0)
cv.destroyAllWindows()

*** Console de processus distant Réinitialisée ***
Training done -----
>>>

```



La reconnaissance de la photo de l'acteur est de 60%.

Essayer les autres images des différents personnages. Il se peut que le taux de réussite ne soit pas aussi bon car le nombre d'images lors de la phase d'apprentissage était limité.

Remarque : la dernière partie présentée dans la vidéo YouTube n'est pas abordée dans ce document.