المحاضرة الأولى

معالجة الصورة - بايثون

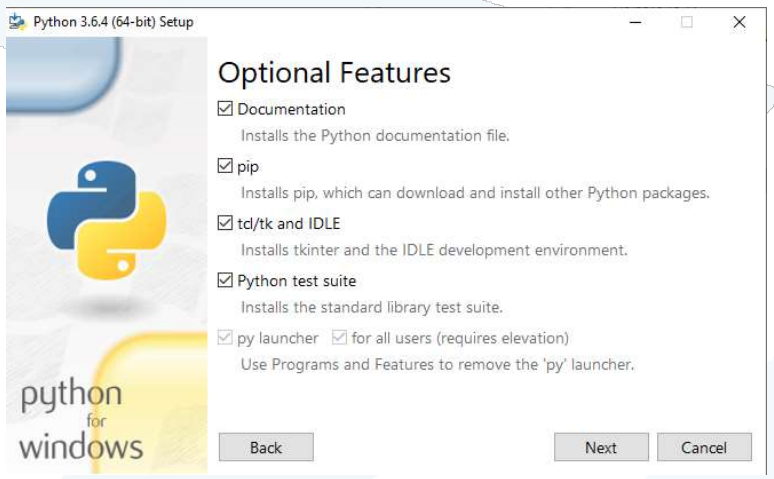
بايثون لغة برمجة عالية المستوي High-Levelبمعني أنها قريبة جداً من لغة البشر مقارنة باللغات منخفضة المستوي Low-Levelوالتي تكون قريبة من لغة الآلة، وهذا يجعلها سهلة الاستخدام والفهم والتعلم.

لدى لغة بايثون حوالي المليون مستخدم، وهي من أوسع لغات البرمجة استخداما، ويشتركُ أغلب مستخدمي بايثون في بعض أسباب وعوامل تدفعهم لاستخدامها نلخصها كما يلي:

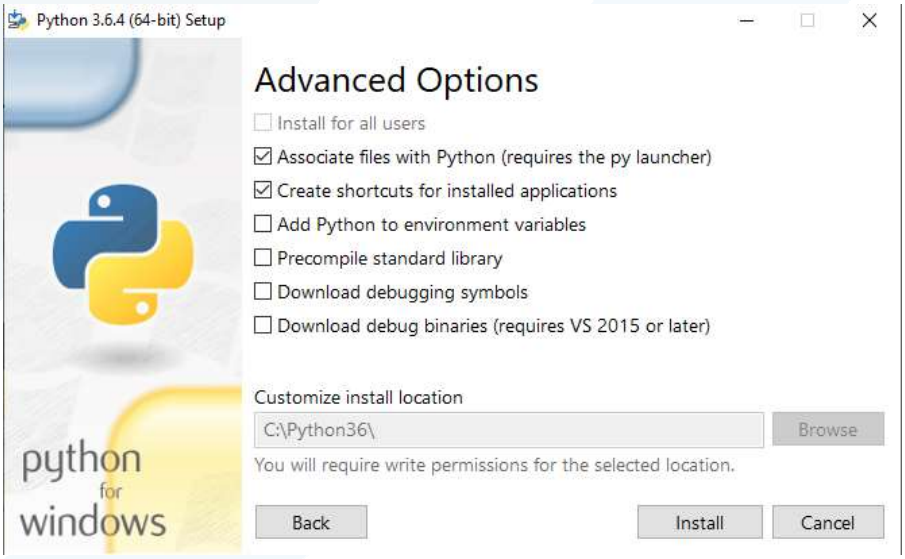
- جودة البرمجيات :Software Quality تتبني فلسفة بايثون نهج الاكتفاء بالحد الأدنى، هذا يعني أنه في نهج بايثون البساطة أفضل من التعقيد، بالتالي فإن بايثون تركز وبشكل كبير على أن يكون بناء الجمل البرمجية بسيطاً وقابلاً للقراءة، وبجانب هذا تمتلك بايثون العديد من الأدوات التي تجعل إعادة استخدام الكود البرمجي شيء سهل، ولأن بايثون تركز علي الجودة بهذا الشكل فمن الطبيعي أن يكون مبرمجي بايثون كذلك.
- إنتاجية المطور :Developer Productivity تعمل بايثون على تعظيمها بشكل كبير عن باقي اللغات مثل سي وجافا وخلافه، فالكود البرمجي في بايثون أقل بمقدار 3 إلى 5 مرات من مثيله في باقي اللغات، وهذا يسهل على المطور كتابة الكود وتصحيحه وصيانته.
- سهولة نقل البرنامج :Program Portability برامج بايثون تعمل بدون تغيير على جميع أنظمة التشغيل مثل ويندوز \لينكس وخلافه، ولهذا لن تحتاج لأي شيء إضافي لتحمل معك البرامج في كل مكان.
- مكتبات الدعم :Support Libraries تأتي بايثون مع مجموعة ضخمة من الوظائف مسبقة الصنع تسمي المكتبة المبدئية، وبجانب هذا يمكن تحظي بايثون بمجموعة ضخمة من مكتبات الطرف الثالث والتي يطورها مجتمع بايثون نفسه في مختلف المجالات.
- التكامل مع باقي اللغات :Component Integration الأكواد البرمجية لبايثون تستطيع وبكل سهولة التواصل مع الأجزاء الأخرى من البرنامج المكتوبة بلغات برمجيه أخري، فمثلاً تستطيع بايثون استدعاء مكتبات برمجية من CوC++ كما تستطيع التكامل مع Java فهي لا تقف وحيدة بمعزل عن باقي اللغات.

تحميل برنامج البايثون:

يوجد العديد من نسخ البايثون المجانية والتي تعمل سواء على نظام Linux أو نظام Windows وعلى كلا البنيتين 32,64bit. يمكن الدخول إلى موقع بايثون وتنزيل النسخة التنفيذية واختيار النسخة المناسبة, سنعمل على النسخة Python3.6.4. عند تنصيب البرنامج يجب أن يتم تحديد مسار أسهل كما في الشكل التالي من أجل تسهيل تنصيب المكتبات كما سنرى. عند تشغيل الملف التنفيذي تظهر النافذة:



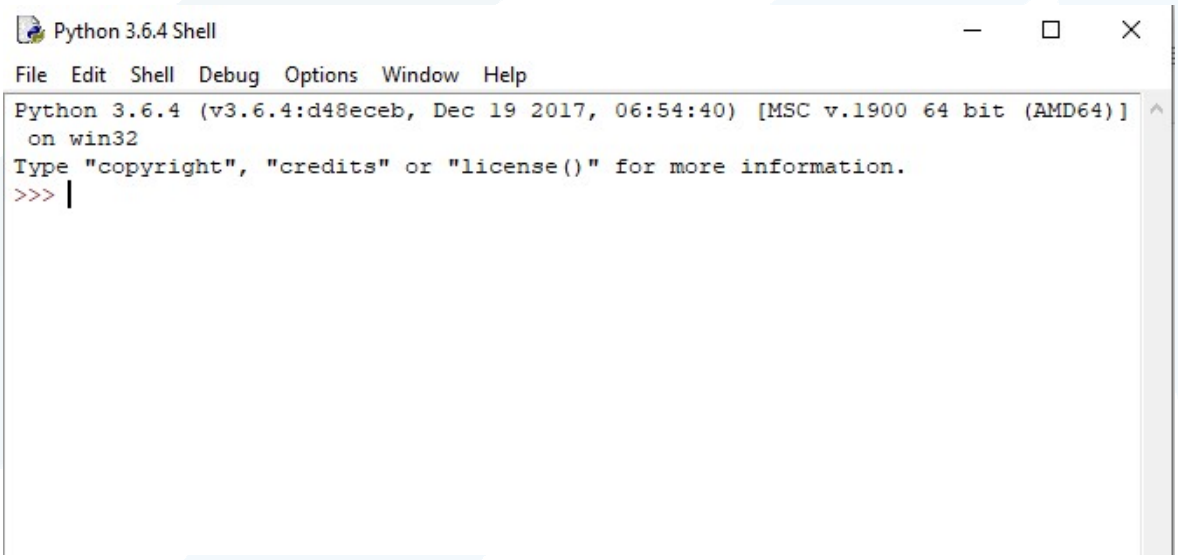
نختار Next:



ضمن خيار Customize install location نحذف المسار المحدد ونكتب المسار c:\Pyhthon36\ وننقر install.

سيتم تنزيل البرنامج في القائمة إبدأ كأي برنامج عادي.

عند تشغيل البرنامج IDLE(Python 3.6 64bit) سنرى الواجهة المبسطة للبرمجية والتي تتضمن الرمز >>> والتي تعني أن البايثون جاهز لتلقي التعليمات كما يمكن من القائمة File اختيار New File من أجل حفظ الكود البرمجي واختباره بشكل دائم.



تتمتع اللغة بنفس البنى العامة للغات البرمجية الأخرى من ناحية مجال الحروف المسموحة وتسلسل العمليات وأولويتها، تبدأ اللغة عادة باستدعاء المكتبات باستخدام التعليمة Import

الشروط الأساسية في بايثون:

- اللغة حساسة لحالة الأحرف Case Sensitive وبالتالي فإن الكلمة note تختلف عن الكلمة Note.
- اسم الصنف Class يبدأ بحرف كبير وإذا كان اسم الصنف يتألف من أكثر من كلمة فيجب أن يكون أول حرف من كل كلمة كبير مثل class First و class FirstPythonCode.
- عند تعريف المتحولات نستخدم أحرف صغيرة واذا كان اسم المتغير أكثر من كلمة نستخدم _ على سبيل المثال average=10 و total_score=2.

تعرف الصورة الرقمية على أنها طريقة لتمثيل الصورة التي تراها العين على جهاز الحاسب, تمثل الصورة بعنصر يسمى البيكسل pixel وهو أقل عنصر ممثل للشاشة وعليه تعرف الشاشات على مقدار البيكسلات الممثلة لها مثل: 1024X768.

تختلف الصور بمجموعة من الخصائص نوردها كما يلي:

أولاً – حسب النوع:

- صورة ثنائية: وتمثل بلونين فقط حيث يمثل الصفر اللون الأسود والواحد اللون الأبيض.
- صورة رمادية: وهنا لدينا صورة ممثلة بطبقة واحدة وتعرف فقط بطولها وعرضها, لدينا مجال كامل من التدرجات الرمادية التي يتراوح مجالها حسب دقة الصورة فإذا كان تمثيل الصورة بالنمط Uint8 فهذا يعني أنه لدينا 256 لون تتراوح من ال0 للأسود وحتى 255 للأبيض وبين هذين الرقمين مجال واسع من التدرجات الرمادية أما إذا مانت التمثيل بالنمط uint16 يتراوح المجال بين 0 للأسود وحتى 65535 للأبيض وبالتالي لدينا مجال أوسع للرماديات.
- صورة ملونة: وهنا لدينا ثلاث طبقات ممثلة بثلاث ألوان RGB(Red,Greeb,Blue) حيث تمثل كل طبقة بتدرج رمادي وعند دمجها وعن طريق تراكب الألوان الثلاثة نحصل على اللون, تتمتع كل طبقة بمزايا التدرج الرمادي السابقة.

ثانياً – حسب نمط البيانات:

- Binary: كما ذكرنا سابقاً إما صفر أو واحد.
- Double: وتتراوح بين الصفر والواحد.
- Uint8: بين 0 و 255.
- Uint16: بين 0 و 65535.

ثالثاً – حسب اللاحقة:

- Png: وهي صورة كاملة الحجم والدقة وذات حجم تخزيني عالي.
- Jpeg: وهي صورة مضغوطة ذات حجم مخفض ودقة عالية.
- Gif: وتمثل الصور المتحركة.

مكتبة OpenCV:

وهي مكتبة مرجعية تتضمن العديد من الخيارات الخاصة بمعالجة الصورة والفيديو كمكتبات جاهزة. لتحميل المكتبة يجب أن يكون هناك اتصال مع الانترنت حيث ندخل إلى شريط الأوامر cmd (يمكن العثور عليه في خيارات البحث المرفقة مع قائمة إبدأ في شريط المهام). عند الدخول إلى النافذة نكتب المسار المحدد للبايثون كما يلي (محدد باللون الأزرق):

موجه الأوامر

```
Microsoft Windows [Version 10.0.17763.253]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Gaya>cd c:\python36\scripts
```

بعد نقر Enter يتم الانتقال إلى المسار المحدد نكتب التعليمة التالية:

موجه الأوامر

```
Microsoft Windows [Version 10.0.17763.253]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Gaya>cd c:\python36\scripts

c:\Python36\Scripts>pip install opencv-python
```

حيث سيتم تحميل المكتبة ودمجها بمسار البايثون تلقائياً.

قراءة صورة :

باستخدام التابع cv2.imread() ولكن يجب اعطاء المسار الكامل للصورة او وضعها بمجلد العمل ويكتب الاسم مع اللاحقة ك string(str) .

المتغير الثاني وهو مؤشر flag يحدد كيفية قراءة الصورة , ويأخذ ثلاث قيم :

- cv2.IMREAD_COLOR
- cv2.IMREAD_GRAYSCALE
- cv2.IMREAD_UNCHANGED

وتأثيرهم بالترتيب:

- اقرأ الصورة بالالوان الكاملة (بدون شفافية)
- اقرأ الصورة بالرمادي (مستوي واحد)
- اقرأ الصور بالالوان الكاملة (مع الشفافية)

ويمكن تمرير 1 ,0.-1 كبديل عن الاعلام حرفياً بالترتيب.

كما المثال التالي :

```
import numpy as np

import cv2

# load color image in gray scale

pic = cv2.imread('image1.png',0)
```

عرض الصورة:

باستخدام التابع : cv2.imshow() نظهر الصورة بالنافذة , واتوماتيكياً تتلائم الصورة مع النافذة , واول متغير هو اسم النافذة , والمتغير الثاني هو الصورة , ويمكنك انشاء اي عدد من النوافذ ولكن كل منها له اسم فريد ..

```
cv2.imshow('image',pic)

cv2.waitKey()

cv2.destroyAllWindows()
```

cv2.waitKey() وهو تابع ينتظر لمدة تساوي متغيره الاول بواحدة ميلي ثانية , حتى يتابع التنفيذ ,او تأتيه ضغطة زر , يردها بمتغير الخرج له , (بصيغة ascii) واذا مررنا له صفراً او عدداً سالباً فسوف يتوقف ولا ينتظر ابداً لاي ضغطة زر ..

cv2.destroyAllWindows() ببساطة اغلق كل النوافذ الموجودة , اما التابع cv2.destroyWindow() فيجب تمرير اسم نافذة محدد لها.

ملاحظة :

اذا اردنا تحجيم اظهار الصورة , لما يناسب الشاشة فعلينا استخدام التابع cv2.namedWindow() نمرر له اسم النافذة المرغوب مع علم له خيارين: cv2.WINDOW_NORMAL اوcv2.WINDOW_AUTOSIZE .

```
cv2.namedWindow('image',cv2.WINODOW_NORMAL)

cv2.imshow('image',pic)

cv2.waitKey(0)

cv2.destroyAllWindows()
```

كتابة صورة :

استخدم التابع cv2.imwrite() لحفظ الصورة .

اول متغير لها هو اسم الملف , والثاني هو رمز الصورة (متغيرها)

```
cv2.imwrite('picgray.png',pic)
```

الملخص:

البرنامج التالي يحمل الصورة ويظهرها ويحفظها بالمستوى الرمادي اذا ضغطت 's' او لا يحفظها ويغلق بضغط ESC.

```
import numpy as np

import cv2

img = cv2.imread('pic5.jpg',0)

cv2.imshow('image',img)

k = cv2.waitKey(0)

if k == 27: # wait for ESC key to exit

cv2.destroyAllWindows()

elif k == ord('s'): # wait for 's' key to save and exit

cv2.imwrite('messigray.png',img)

        cv2.destroyAllWindows()
```

باستخدام Matplotlib:

تتيح هذه المكتبة امكانيات افضل لاظهار الصور وحفظها كما يلي:

```
import numpy as np

import cv2

from matplotlib import pyplot as plt

img = cv2.imread('pic.jpg',0)

plt.imshow(img, cmap = 'gray', interpolation = 'bicubic')
```

```
plt.xticks([]), plt.yticks([]) # to hide tick values on X and Y axis

plt.show()
```

ملاحظة :

عند تحميل الصور عبر opencv نحصل على صور بصيغة BGR ولكن عند اظهارها ب Matplotlib يتم اعتبارها RGB (اي يكون الاحمر ازرق والعكس) لذلك يجب مراعاة هذا الوضع والانتباه لتبدبل الالوان قبل اظهارها عبر ال Matplotlib

التعامل مع الكاميرا والفيديو:

احياناً نحتاج لاخذ الصور مباشرة من الكاميرا, ولذلك سنقوم هنا بقراءة الفيديو ومن ثم نحول الاطارات للمستوي الرمادي ونعرضه كمهمة بسيطة للبدء. ولالتقاط فيديو تحتاج لجسم يدعى

VideoCapture

ودخله اما ترتيب الاداة او اسم ملف الفيديو , وغالباً الكاميرا الوحيدة ستملك الترتيب 0 او -1 وبعدها يمكنك التقاط كل اطار بإطاره ولكن لا تنسى بالنهاية افلات جسم الالتقاط.

```
import cv2

import numpy as np

cap = cv2.VideoCapture(1)

while(True):

  # Capture frame-by-frame

  ret, frame = cap.read()

  # print frame

  # convrt 2 gray

  gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)

  # display the result (try in page)

  cv2.imshow('frame',gray)

  if cv2.waitKey(1) == ord('q'):
```

```
        # in 64-bit machine add: & 0xFF above
        break
    # at the end release every thing:
    cap.release()
    cv2.destroyAllWindows()
```

يعطي عددا بولانياً , اذا ما كان الاطار مقروءا بشكل صحيح , ولذلك يمكن معرفة نهاية الفيديو من ذلك , cap.read() اذا اعطى False ويمكنك استخدام cap.isOpened() .

كما يمكنك ايضاً الحصول على خصائص الفيديو من خلال التعليمة

```
cap.get(ProbId)
```

حيث نمرر رقم الخاصية ذو المجال من 0-18 كما نلاحظ ان بعض الخاصيات لديها قابلية القراءة والكتابة ايضاً عبر

```
cap.set(probId,value).
```

مثلاً الطول والعرض للفيديو , نلاحظ انه قد يمكن تغييرهما..

عمليات التدوير:

تعتمد عمليات التدوير على تحويل الصورة إلى مصفوفة من ثم تحديد مركز التدوير وبعد ذلك يتم القيام بعمليات التدوير.

```
import cv2
# read image as grey scale
img = cv2.imread('/home/arjun/Desktop/logos/python.png')
# get image height, width
(h, w) = img.shape[:2
        ]
# calculate the center of the image
center = (w / 2, h / 2)
```

```
angle90 = 90

angle180 = 180

angle270 = 270

scale = 1.0

# Perform the counter clockwise rotation holding at the center

# 90 degrees

M = cv2.getRotationMatrix2D(center, angle90, scale)

rotated90 = cv2.warpAffine(img, M, (h, w))

# 180 degrees

M = cv2.getRotationMatrix2D(center, angle180, scale)

rotated180 = cv2.warpAffine(img, M, (w, h))

# 270 degrees

M = cv2.getRotationMatrix2D(center, angle270, scale)

rotated270 = cv2.warpAffine(img, M, (h, w))

cv2.imshow('Original Image',img)

cv2.waitKey(0) # waits until a key is pressed

cv2.destroyAllWindows() # destroys the window showing image

cv2.imshow('Image rotated by 90 degrees',rotated90)

cv2.waitKey(0) # waits until a key is pressed

cv2.destroyAllWindows() # destroys the window showing image

cv2.imshow('Image rotated by 180 degrees',rotated180)

cv2.waitKey(0) # waits until a key is pressed

cv2.destroyAllWindows() # destroys the window showing image
```

```
cv2.imshow('Image rotated by 270 degrees',rotated270)

cv2.waitKey(0) # waits until a key is pressed

cv2.destroyAllWindows() # destroys the window showing image
```

عمليات القص:

```
import numpy as np

import cv2

image = cv2.imread('download.jpg')

y=0

x=0

h=100

w=200

crop = image[y:y+h, x:x+w]

cv2.imshow('Image', crop)

cv2.waitKey(0)
```

عمليات التحجيم:

التصغير:

```
import cv2

img = cv2.imread('/home/img/python.png', cv2.IMREAD_UNCHANGED)

print('Original Dimensions : ',img.shape)

scale_percent = 60 # percent of original size

width = int(img.shape[1] * scale_percent / 100)

height = int(img.shape[0] * scale_percent / 100)
```

```
dim = (width, height)
# resize image
resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
print('Resized Dimensions : ',resized.shape)
cv2.imshow("Resized image", resized)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

التكبير:

```
import cv2
img = cv2.imread('/home/img/python.png', cv2.IMREAD_UNCHANGED)
print('Original Dimensions : ',img.shape)
scale_percent = 220 # percent of original size
width = int(img.shape[1] * scale_percent / 100)
height = int(img.shape[0] * scale_percent / 100)
dim = (width, height)
# resize image
resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
print('Resized Dimensions : ',resized.shape)
cv2.imshow("Resized image", resized)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

التكبير أفقياً:

```
import cv2
```

```
img = cv2.imread('/home/img/python.png', cv2.IMREAD_UNCHANGED)
print('Original Dimensions : ',img.shape)
width = 440
height = img.shape[0] # keep original height
dim = (width, height)
# resize image
resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
print('Resized Dimensions : ',resized.shape)
cv2.imshow("Resized image", resized)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

التكبير شاقولياً:

```
import cv2
img = cv2.imread('/home/img/python.png', cv2.IMREAD_UNCHANGED)
print('Original Dimensions : ',img.shape)
width = img.shape[1] # keep original width
height = 440
dim = (width, height)
# resize image
resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
print('Resized Dimensions : ',resized.shape)
```

```
cv2.imshow("Resized image", resized)

cv2.waitKey(0)

cv2.destroyAllWindows()
```

هستوغرام الصورة:

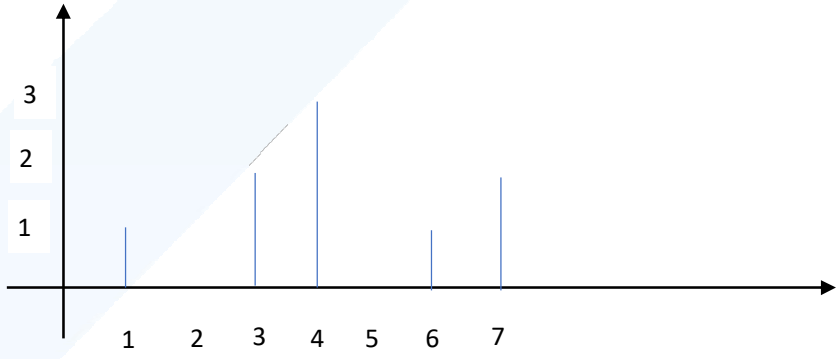
الهستوغرام عبارة عن مخطط يعبر عن توزع الرماديات في الصورة حيث يمثل بمحورين العمودي يعبر عن تكرار توارد الرماديات في صورة ما والأفقي يمثل قيمة الرماديات.

على سبيل المثال إذا كانت الصورة ممثلة بالقيمة 3bit عندها يكون مجال الرماديات في الصورة محصوراً بين 0 و 7 وهي القيم الموجودة في المحور الأفقي.

بفرض لدينا الصورة التالية:

3	3	1
4	4	4
7	7	6

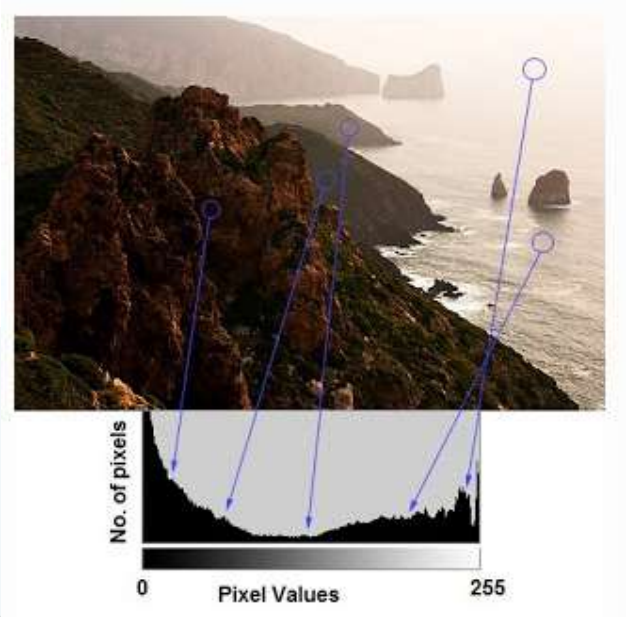
وبالتالي الهستوغرام المعبر عن الصورة:



يوجد مجموعة مفاهيم يعبر فيها هذا الهستوغرام عن وضع الصورة كما يلي:

- إذا كان الهستوغرام متجمع باتجاه القيم العظمى فالصورة عالية السطوع.
- إذا كان الهستوغرام مائل للقيم الدنيا فالصورة منخفضة السطوع (عاتمة).

- إذا كان المخطط في المنتصف فالصورة ذات مجال ضيق من الرماديات.
- إذا كان المخطط موزع على كامل المجال فالصورة ذات مجال واسع من الرماديات.



يمكن حساب هذا المخطط سواء اعتماداً على OpenCV أو Numpy كما يلي:

باستخدام OpenCV:

ويتم ذلك باستخدام التعليمة التالية:

cv2.calcHist(images, channels, mask, histSize, ranges[, hist[, accumulate]])

تتضمن التعليمة القيم التالية:

المحدد	الوصف
images	الصورة الدخل وهي من نوع uint8 أو float32 ويجب إدخال الصورة بالشكل [img]
channels	وكذلك يجب أن تدخل على شكل أقواس فإذا كانت الصورة رمادية تدخل على شكل [0] وإذا كانت ملونة فإننا ندخل [0],[1],[2] وذلك حسب الطبقة اللونية.

mask	تحديد مجال حساب الهستوغرام بحيث ندخل None لحساب هستوغرام كامل الصورة وإلا فيجب إدخال قناع محدد.
histSize	وهنا نحدد الرماديات التي نريد حساب الهستوغرام لها وإذا أردنا جميع الرماديات نضع [256]
ranges	يحدد قيم الرماديات التي نقوم بالبحث عنها عملياً هو بالمجال [0,256]

مثال:

```
img = cv2.imread('home.jpg',0)

hist = cv2.calcHist([img],[0],None,[256],[0,256])
```

باستخدام numpy:

```
hist,bins = np.histogram(img.ravel(),256,[0,256])
```

طباعة مخطط الهستوغرام:

باستخدام matplotlib:

```
import cv2

import numpy as np

from matplotlib import pyplot as plt

img = cv2.imread('home.jpg',0)

plt.hist(img.ravel(),256,[0,256]); plt.show()
```

أو باستخدام التابع التقليدي:

```
import cv2

import numpy as np

from matplotlib import pyplot as plt
```

```
img = cv2.imread('home.jpg')
color = ('b','g','r')
for i,col in enumerate(color):
    histr = cv2.calcHist([img],[i],None,[256],[0,256])
    plt.plot(histr,color = col)
    plt.xlim([0,256])
plt.show()
```

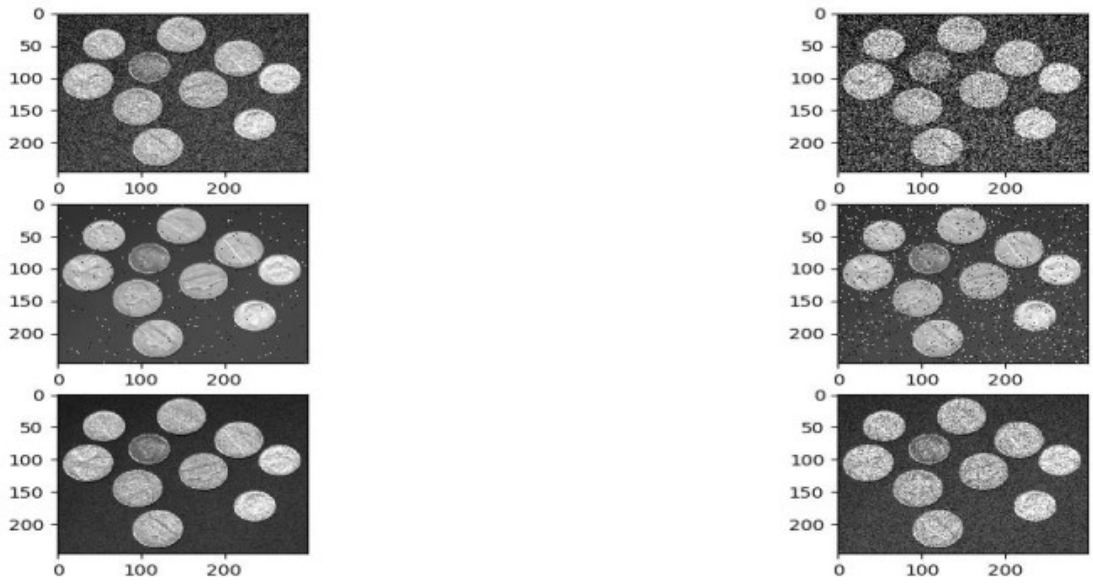
الهستوغرام باستخدام القناع:

```
img = cv2.imread('home.jpg',0)
# create a mask
mask = np.zeros(img.shape[:2], np.uint8)
mask[100:300, 100:400] = 255
masked_img = cv2.bitwise_and(img,img,mask = mask)
# Calculate histogram with mask and without mask
# Check third argument for mask
hist_full = cv2.calcHist([img],[0],None,[256],[0,256])
hist_mask = cv2.calcHist([img],[0],mask,[256],[0,256])
plt.subplot(221), plt.imshow(img, 'gray')
plt.subplot(222), plt.imshow(mask,'gray')
plt.subplot(223), plt.imshow(masked_img, 'gray')
plt.subplot(224), plt.plot(hist_full), plt.plot(hist_mask)
plt.xlim([0,256])
plt.show()
```

الترشيح في المجال المكاني:

إضافة أنواع الضجيج للصورة الرقمية:

```
import skimage
import matplotlib.pyplot as plt
import cv2
img=cv2.imread('coins.png',0)/255.0
img1=skimage.util.random_noise(img,"gaussian",mean=0,var=0.01)
img11=skimage.util.random_noise(img,"gaussian",mean=0,var=0.05)
img2=skimage.util.random_noise(img,'s&p',amount=0.01)
img22=skimage.util.random_noise(img,'s&p',amount=0.05)
img3=skimage.util.random_noise(img,'speckle',mean=0,var=0.01)
img33=skimage.util.random_noise(img,'speckle',mean=0,var=0.05)
plt.figure(1)
plt.subplot(321),plt.imshow(img1,cmap='gray')
plt.subplot(322),plt.imshow(img11,cmap='gray')
plt.subplot(323),plt.imshow(img2,cmap='gray')
plt.subplot(324),plt.imshow(img22,cmap='gray')
plt.subplot(325),plt.imshow(img3,cmap='gray')
plt.subplot(326),plt.imshow(img33,cmap='gray')
plt.show()
```



إزالة الضجيج الغوصي:

من أشهر المرشحات المستخدمة لإزالة الضجيج الغوصي هو المرشح الغوصي Gaussian filter و المرشح المتوسط mean filterاو average filter التي تصنف بأنها مرشحات الخطية والمرشح bilateralfilter الذي يصنف بأنه مرشح لا خطي

1- المرشح المتوسط Mean Filter

يقوم هذا المرشح بأخذ القيمة المتوسطة للبكسل ومجاوارته ضمن جوار محدد ويستبدل قيمة البكسل بهذه القيمة المتوسطة.

يمكن بناء مرشح mean بقناع بحجم 3*3 كمايلي

```
h1=np.ones((3,3),np.float32)/9
```

ومرشح متوسط بقناع ذو حجم 5*5

```
h2=np.ones((5,5),np.float32)/25
```

ومن ثم تطبيق القناع على الصورة باستخدام الأمر

```
imgh1=cv2.filter2D(img,-1,h1)
```

ويمكن كتابة أمر بناء مرشح متوسط بحجم 3*3وتطبيقه بشكل مباشر على الصورة كما يلي

```
imgm1=cv2.blur(img,(3,3))
```

يتم تكرار عملية تطبيق القناع على بقية بكسلات الصورة ومجاوارت كل منها بشكل التفافي حتى الوصول إلى آخر بكسل .

مثال: احسب ناتج تطبيق المرشح المتوسط بقناع 5*5 على البكسل ذو الاحداثيات (1,1) للصورة التالية:

قناع المرشح أو نافذة المرشح أو جوار البكسل

100	50	10	5	60
10	20	15	55	66
70	60	200	26	100
70	55	66	20	20
20	26	100	20	100

$$New_{value} of\ pixel \quad \frac{100+50+10+10+20+15+70+60+100}{9} = 59.44 \approx 60$$

هنا نلاحظ أن قيمة البكسل (1,1) ارتفعت من القيمة 20 إلى القيمة 60

هناك نماذج أخرى من مرشح المتوسط تتميز بأنها أكثر ديناميكية من القناعين السابقين حيث يتم إعطاء وزن لكل بكسل ضمن جوار القناع وكلما زاد البعد عن مركز القناع تقل قيمة الوزن التي سيضرب بها البكسل وفيما يلي بعض منها

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}, \quad \frac{1}{256}\begin{bmatrix}1 & 4 & 6 & 4 & 1\\ 4 & 16 & 24 & 16 & 4\\ 6 & 24 & 36 & 24 & 6\\ 4 & 16 & 24 & 16 & 4\\ 1 & 4 & 6 & 4 & 1\end{bmatrix}$$

2- المرشح الغوصي Gaussian Filter

يعتبر هذا المرشح أفضل من المتوسط في موضوع ترشيح الضجيج الغوصي كونه يعتمد نموذج توزع غوص ذاته الذي يعتمده الضجيج الغوصي يتم تطبيقه وفق التعليمة التالية

```
imgg1=cv2.GaussianBlur(img,(m,m),sigma)
```

. حيث m هي حجم نافذة المرشح (قناع المرشح) أما sigma فهي الانحراف المعياري للمرشح وكلما ازدت تزداد شدة الترشيح علما أنه عندما تكون sigma=0 فإن المرشح يسلك سلوك مرشح متوسط متكيف

3- المرشح :bilateralfilter

هو مرشح لا خطي وله فعالية عالية في حذف الضجيج مع المحافظة على الحواف في الصورة ,ولذلك تكون عملية الترشيح بطيئة مقارنة مع غيره من المرشحات يتم تطبيقه وفق مايلي

```
imgw=cv2.bilateralFilter(img,d,sigmacolor,sigmaspace)
```

نفذ الكود التالي

```
#linearfilter1.py
import numpy as np
import matplotlib.pyplot as plt
import cv2

h1=np.ones((3,3),np.float32)/9
h2=np.ones((5,5),np.float32)/25
h3=np.array([[1,2,1],[2,4,2],[1,2,1]])/16

h4=np.array([[1 ,4,6,4, 1],[4 ,16, 24 ,16, 4],[6 ,24, 36, 24 ,6],[4 ,16, 24 ,16,
4],[1 ,4,6,4, 1]])/256
img=cv2.imread('coins1.png',0)
imgh1=cv2.filter2D(img,-1,h1)
imgh2=cv2.filter2D(img,-1,h2)
imgh3=cv2.filter2D(img,-1,h3)
imgh4=cv2.filter2D(img,-1,h4)
imgm1=cv2.blur(img,(3,3))
imgm2=cv2.blur(img,(5,5))
imgg1=cv2.GaussianBlur(img,(3,3),0)
imgg2=cv2.GaussianBlur(img,(5,5),0)
imgw=cv2.bilateralFilter(img,3,75,75)
print(imgh1[0:3,0:3])
print(imgm1[0:3,0:3])
print(imgh3[0:3,0:3])
print(imgg1[0:3,0:3])
cv2.imshow('img',img)
cv2.waitKey(2000)
cv2.imshow('imgh1',imgh1)
```

إزالة ضجيج salt & pepper:

تعتبر المرشحات الخطية غير فعالة في مجال إزالة هذا النوع من الضجيج كونها تعتمد على جميع قيم الجوار بما فيها قيمة البكسل ذاته لإزالة الضجيج، مما يعني أن البكسل الضجيجي سيدخل في عملية التعديل وسيؤثر على الناتج. أما المرشحات اللاخطية فهي تعتمد على استبدال قيمة البكسل بأحد البكسلات الموجودة في جواره. ولعل أشهر هذه المرشحات المرشح الوسيط الذي يقوم بأخذ جوار معين للبكسل المدورس ومن ثم ترتيبه وأخذ العنصر الوسيط

مثال : طبق مرشح الوسيط بحجم 3*3 على البكسل (1,1) في الصورة التالية :

100	50	70	5	60
60	0	100	55	66
70	60	200	26	100
70	55	66	20	20
20	26	100	20	100

نرتّب الجوار تصاعدياً: (0-50-60-60-70-70-100-100-200) الوسيط هو العنصر 70 بالتالي ستصبح قيمة البكسل 70 بدلاً من 0 وسيصبح متجانساً مع البكسلات المحيطة به.

يطبق مرشح الوسيط باستخدام التعليمة التالية :

```
img3=cv2.medianBlur(img,m)
```

نفذ الكود التالي

```
#replacenoisefilt.py
import cv2
img=cv2.imread('woman.jpg',0)
img1=cv2.blur(img,(3,3))
img2=cv2.bilateralFilter(img,5,75,75)
img3=cv2.medianBlur(img,3)
img4=cv2.medianBlur(img,5)
```

العمليات المورفولوجية:

تهدف عملية الحت إلى تقليص حجم العناصر في الصورة من أجل إظهار الهياكل الاساسية حيث يتم حذف بيكسلات من الصورة وفقاً لبنية عنصر يسمى معامل البناء (SE(structural element.

لتحويلات المورفولوجية , هي عمليات بسيطة على الصور الثنائية عادة , ولها دخلين هما الصورة الاصلية والعنصر التركيبي الذي يحدد طبيعة العملية والعمليات الاساسية هي

Erosion , Dilation

ثم يأتي بعدها كتركيب لها , الفتح والاغلاق والتدرج وسنطبق ما سبق ,واحدا عقب الاخر على الصورة التالية

In [2]:
```
from IPython.display import Image
Image('j.png')
```
Out[2]:

Erosion

هذا التحويل يحاول تنعيم سطح الامامية , التي دائما بيضاء , وذلك بجعل قيمة العنصر 1 فقط في حالة العناصر المحيطة به كلها كذلك تحت القناع.1

وبذلك بالاعتماد على حجم القناع تتحدد الثخانة للناتج , وهذا يفيد بازالة الضجيج كالنقاط البيضاء الصغيرة او فصل جسمين متصلين خطأً, والتالي يبين قناعاً كله واحدات يطبق ما سبق

In [10]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('j.png',0)
kernel = np.ones((5,5),np.uint8)
erosion = cv2.erode(img,kernel,iterations = 1)

plt.imshow(erosion,cmap = 'gray')
plt.xticks([]) , plt.yticks([]) , plt.show()
None
```

Dilation

وهذا يقابل السابق , , فهنا البكسل يأخذ القيمة 1 اذا كان واحد من عناصره تحت القناع 1 على الاقل, وهكذا فهو يزيد حجم المنطقة البيضاء , وبالعادة بالحالات الخاصة بازالة الضجيج يتبع هذا ال

Erosion

لان الاخير يزيل الضجيج ولكنه ايضاً يقلل مساحة الجسم , ولذلك نطبق التحويل

Dilation

وهو يفيد ايضاً بجمع اجزاء الجسم المتكسرة , كالتالي

In [9]:
```
dilation = cv2.dilate(img,kernel,iterations = 1)

plt.imshow(dilation,cmap = 'gray')
plt.xticks([]) , plt.yticks([]) , plt.show()
None
```

:الفتح

الفتح هو اسم اخر لتطبيق

Erosion ,then ,Dilation

وهذا يفيد بازالة الضجيج كما شرحنا اعلاه , ويتم عبر التابع

cv2.morphologyEx()

In [14]:
```
import random
h , w = img.shape
# add noise
for x in range(15):
    cv2.circle(img,(random.randint(0,w),random.randint(0,h)),2,
               (255,255,255),-1)

opening = cv2.morphologyEx(img, cv2.MORPH_OPEN, kernel)
```

```
res = np.hstack([img,opening])
plt.imshow(res,cmap = 'gray')
plt.xticks([]) , plt.yticks([]) , plt.show()
None
```

الاغلاق:

وهو عكس الفتح , اي

Dilation ,then , Erosion

وهو يفيد باغلاق الثقوب السوداء في الجسم , او النقاط السوداء

In [15]:

```
img = cv2.imread('j.png',0)

# add noise
for x in range(100):
    cv2.circle(img,(random.randint(0,w),random.randint(0,h)),1,
               (0,0,0),-1)

closing = cv2.morphologyEx(img, cv2.MORPH_CLOSE, kernel)

res = np.hstack([img,closing])
plt.imshow(res,cmap = 'gray')
plt.xticks([]) , plt.yticks([]) , plt.show()
None
```

التدرج الموفولوجي :

وهو الفرق بين ال

Dilation and the Erosion

لصورة , والنتيجة ستبدو كحدود للجسم ,كالتالي

In [16]:

```
img = cv2.imread('j.png',0)

gradient = cv2.morphologyEx(img, cv2.MORPH_GRADIENT, kernel)

res = np.hstack([img,gradient])
plt.imshow(res,cmap = 'gray')
plt.xticks([]) , plt.yticks([]) , plt.show()
None
```