Εργαστήριο 6
Τεχνητής Όρασης

Κώστας Μαριάς

Σκοπός Εργαστηρίου

Σκοπός Εργαστηρίου:

- Να μάθουμε να οξύνουμε μια εικόνα με Laplacian mask
- Να μάθουμε να οξύνουμε μια εικόνα με μέθοδο unsharp
- Να μάθουμε να βρίσκουμε το πλάτος του διανύσματος της κλίσης μιας εικόνας (Sobel method) για μή γραμμική όξυνση

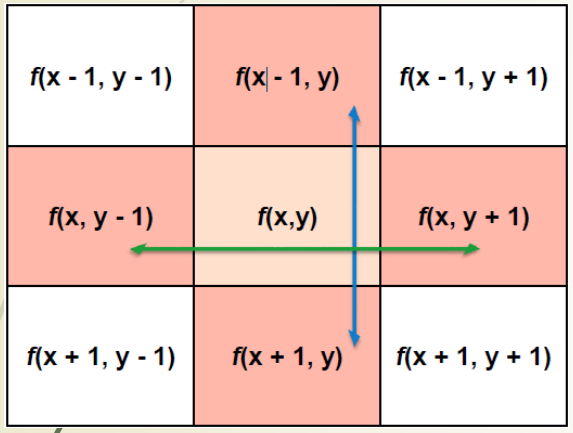
Laplacian mask

- Η Λαπλασιανή μάσκα ως τελεστής παραγώγου ενισχύει ασυνέχειες έντασης στην εικόνα (π.χ. ακμές) ενώ παράλληλα αποδυναμώνει περιοχές όπου οι εντάσεις μεταβάλλονται αργά.
- Το αποτέλεσμα είναι να έχουμε εικόνα με γκρίζες αποχρώσεις-γραμμές στις ακμές και ασυνέχειες της εικόνας στο προσκήνιο, ενώ το υπόλοιπο εμφανίζεται ως σκοτεινό υπόβαθρο χωρίς ιδιαίτερα χαρακτηριστικά.
- Τα χαρακτηριστικά του υποβάθρου μπορούν να «ανακτηθούν», μαζί με την επίδραση-όξυνση της Λαμπλασιανής απλά προσθέτοντας (ή αφαιρώντας) την Λαπλασσιανή εικόνα στην (από την) αρχική

Laplacian mask

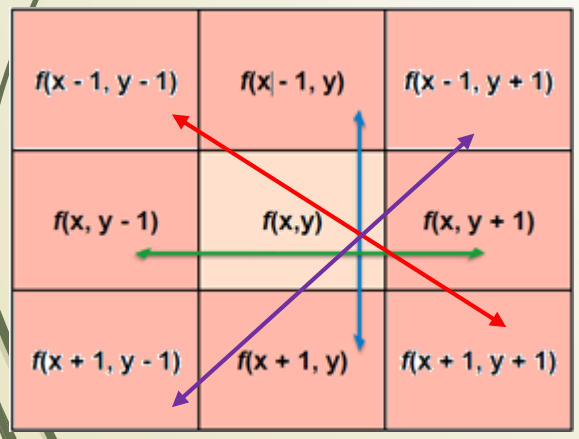
- Ο τύπος της Λαπλασιανής μάσκας ορίζεται ως εξής:

$$\nabla^2 f(x,y) = \frac{\theta^2 f(x,y)}{\theta x^2} + \frac{\theta^2 f(x,y)}{\theta y^2}$$



0	1	0
1	-4	1
0	1	0

$$= f(x+1,y) + f(x+1,y) + f(x,y-1) + f(x,y+1) - \mathbf{4}\boldsymbol{f}(\boldsymbol{x},\boldsymbol{y})$$



1	1	1
1	-8	1
1	1	1

$$= f(x+1,y) + f(x+1,y) + f(x,y-1) + f(x,y+1) + f(x+1,y+1) + f(x-1,y-1) + f(x+1,y-1) + f(x-1,y+1) \text{ -8 } \boldsymbol{f}(\boldsymbol{x},\boldsymbol{y})$$

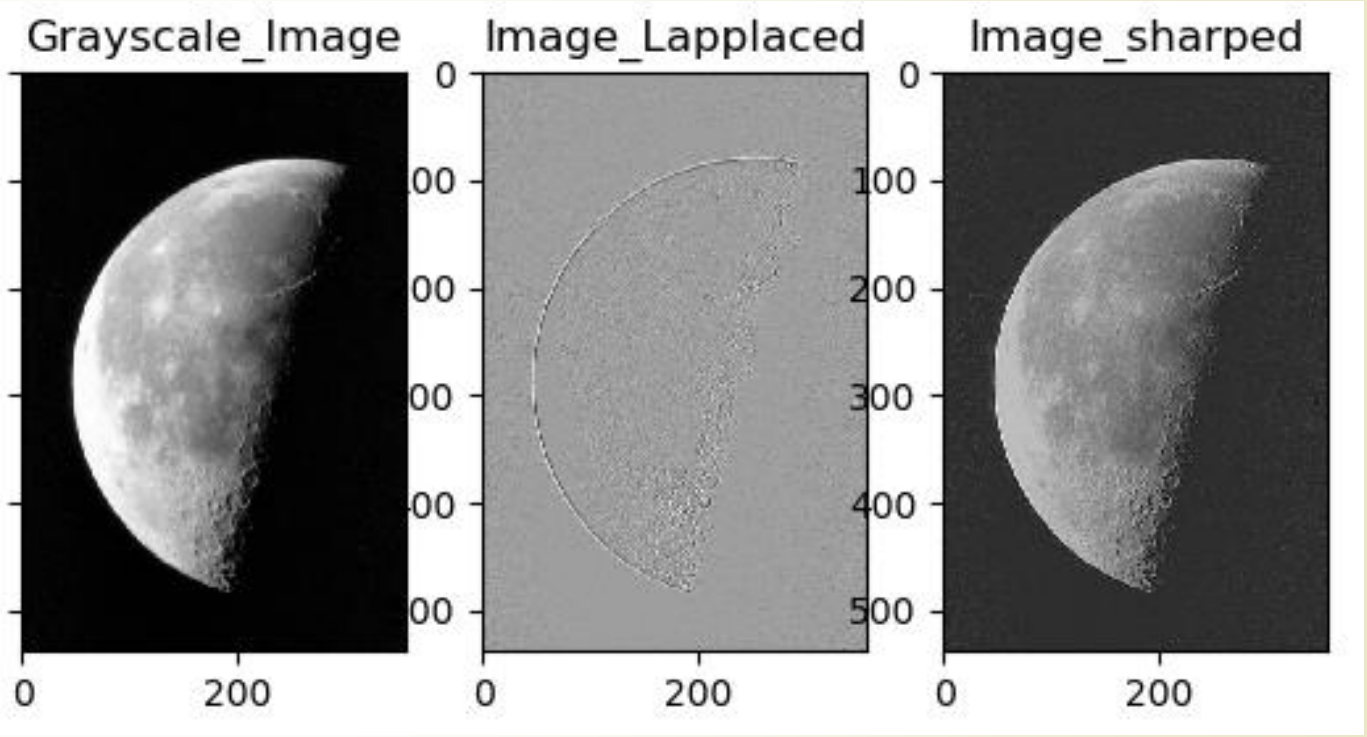
Sharpening with Laplacian mask

$$g(x,y) = f(x,y) + c \cdot \nabla^2 f(x,y)$$

- $\boldsymbol{f(x,y)}$ **και** $\boldsymbol{g(x,y)}$ είναι οι εικόνες εισόδου και η βελτιωμένη εικόνα με όξυνση αντίστοιχα.
- Αν ο ορισμός που χρησιμοποιείται για τη Λαπλασιανή έχει αρνητικό συντελεστή κέντρο, τότε αφαιρούμε, αντί να προσθέσουμε τη Laplacian εικόνα για να έχουμε αποτέλεσμα όξυνσης ⟶ c=−1

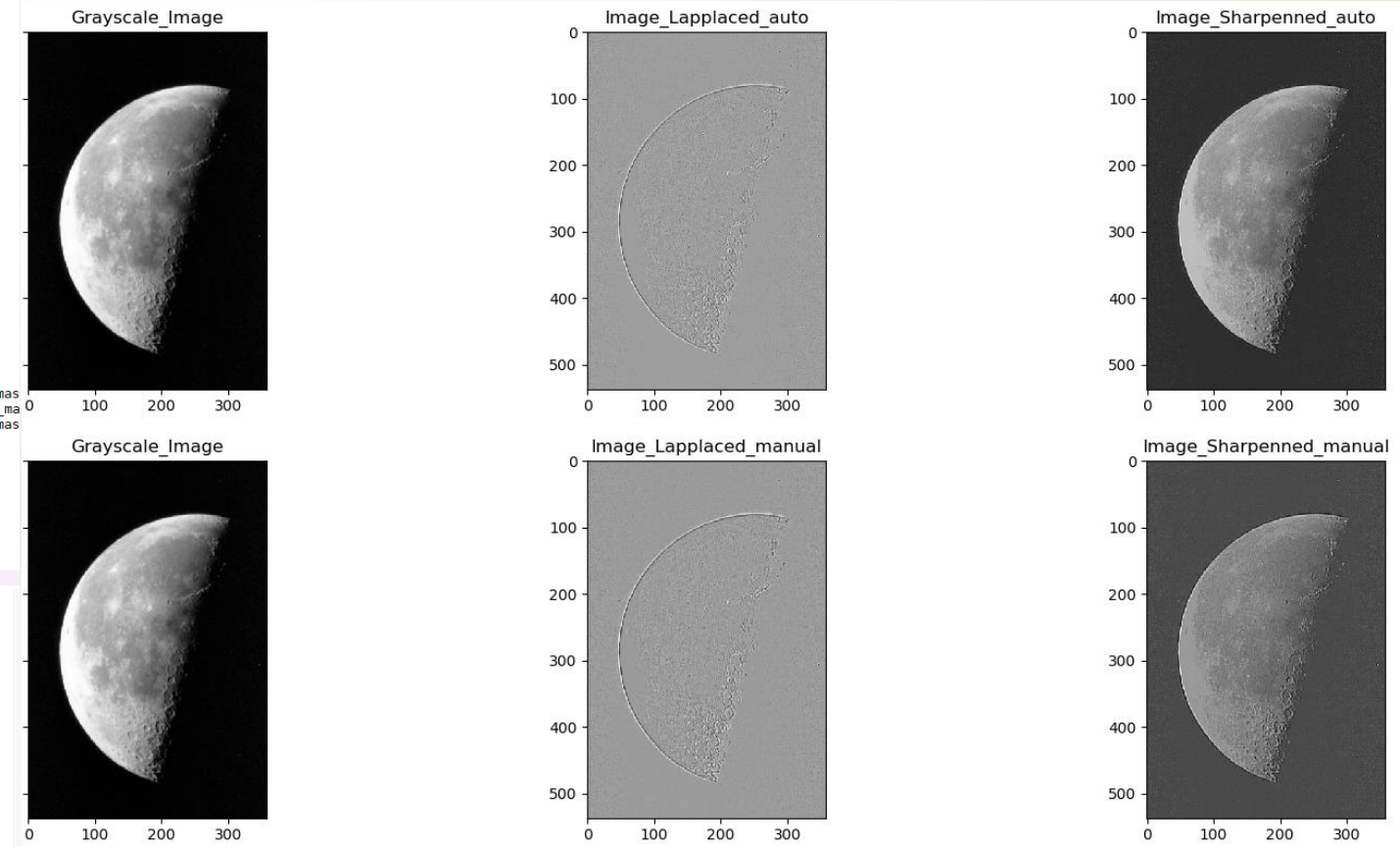
Sharpening with Laplacian mask #Auto


```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 import matplotlib.image as mpimg
4 import Image_formats_and_conversions as tools
5 import scipy.ndimage
6
7 Grayscale_Image = mpimg.imread('moon.tiff')
8
9 Grayscale_Image= np.asarray(Grayscale_Image, dtype = np.float64 )
10
11 #Image_Lapplaced=scipy.ndimage.gaussian_laplace(Grayscale_Image,sigma=0.6)
12 Image_Lapplaced=scipy.ndimage.filters.laplace(Grayscale_Image)
13
14 Image_sharped=Grayscale_Image-Image_Lapplaced
15 Image_sharped=tools.float_to_uint8(Image_sharped)
16
17 figure1=plt.figure(1)
18
19 subplot1=figure1.add_subplot(1,3,1)
20 plt.imshow(Grayscale_Image,cmap="gray")
21 subplot1.set_title('Grayscale_Image')
22
23 subplot1=figure1.add_subplot(1,3,2)
24 plt.imshow(Image_Lapplaced,cmap="gray")
25 subplot1.set_title('Image_Lapplaced')
26
27 subplot1=figure1.add_subplot(1,3,3)
28 plt.imshow(Image_sharped,cmap="gray")
29 subplot1.set_title('Image_sharped')
30
```



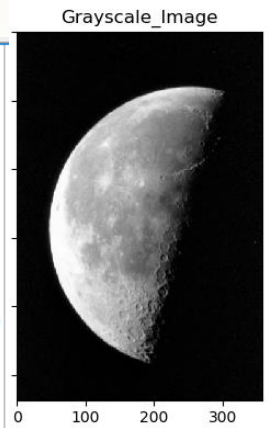
Sharpening with Laplacian mask #Manual_1

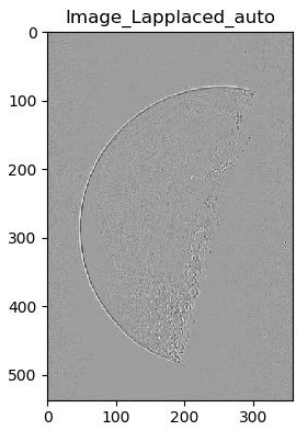

```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import Image_formats_and_conversions as tools
import scipy.ndimage


Grayscale_Image = mpimg.imread('moon.tiff')

Grayscale_Image= np.asarray(Grayscale_Image, dtype = np.float64 )

#Image_Lapplaced_auto=scipy.ndimage.gaussian_laplace(Grayscale_Image,sigma=0.6)
Image_Lapplaced_auto=scipy.ndimage.filters.laplace(Grayscale_Image)

Image_Sharpenned_auto=Grayscale_Image-Image_Lapplaced_auto
Image_Sharpenned_auto=tools.float_to_uint8(Image_Sharpenned_auto)

#-----------------------------------------------------------------------------------------------------------
mask_manual=np.ones((3,3))
mask_manual[1,1]=-8
#mask_manual=[[0,1,0],[1,-4,1],[0,1,0]]
#mask_manual= np.asarray(mask_manual, dtype = np.int32 )



rows=len(Grayscale_Image)
columns=len(Grayscale_Image[0])

Image_Lapplaced_manual=np.zeros((rows,columns))

for i in range(1,(rows-1)):
    for k in range(1,(columns-1)):
        Image_Lapplaced_manual[i,k]= mask_manual[0,0]*Grayscale_Image[i-1,k-1]+mask_manual[0,1]*Grayscale_Image[i-1,k]+mas
                                    +mask_manual[1,0]*Grayscale_Image[i,k-1]+mask_manual[1,1]*Grayscale_Image[i,k]+mask_ma
                                    +mask_manual[2,0]*Grayscale_Image[i+1,k-1]+mask_manual[2,1]*Grayscale_Image[i+1,k]+mas

Image_Sharpenned_manual=Grayscale_Image-Image_Lapplaced_manual
Image_Sharpenned_manual=tools.float_to_uint8(Image_Sharpenned_manual)

figure1=plt.figure(1)

subplot1=figure1.add_subplot(2,3,1)
plt.imshow(Grayscale_Image,cmap="gray")
subplot1.set_title('Grayscale_Image')

subplot1=figure1.add_subplot(2,3,2)
plt.imshow(Image_Lapplaced_auto,cmap="gray")
subplot1.set_title('Image_Lapplaced_auto')

subplot1=figure1.add_subplot(2,3,3)
plt.imshow(Image_Sharpenned_auto,cmap="gray")
subplot1.set_title('Image_Sharpenned_auto')

subplot1=figure1.add_subplot(2,3,4)
plt.imshow(Grayscale_Image,cmap="gray")
subplot1.set_title('Grayscale_Image')

subplot1=figure1.add_subplot(2,3,5)
plt.imshow(Image_Lapplaced_manual,cmap="gray")
subplot1.set_title('Image_Lapplaced_manual')

subplot1=figure1.add_subplot(2,3,6)
plt.imshow(Image_Sharpenned_manual,cmap="gray")
subplot1.set_title('Image_Sharpenned_manual')
```

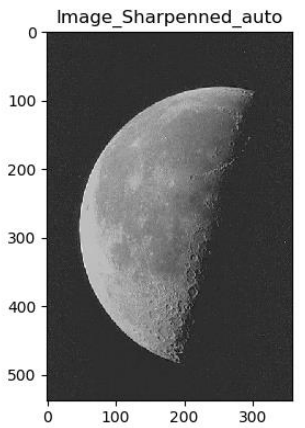


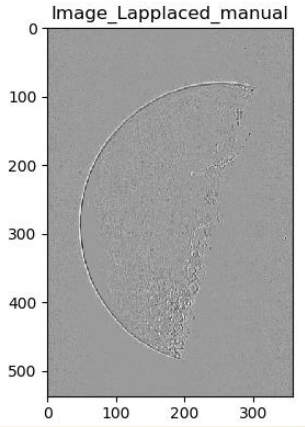
Sharpening with Laplacian mask #Manual_2

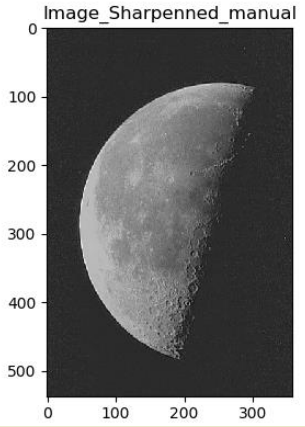

```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import Image_formats_and_conversions as tools
import scipy.ndimage


Grayscale_Image = mpimg.imread('moon.tiff')

Grayscale_Image= np.asarray(Grayscale_Image, dtype = np.float64 )

#Image_Lapplaced_auto=scipy.ndimage.gaussian_laplace(Grayscale_Image,sigma=0.6)
Image_Lapplaced_auto=scipy.ndimage.filters.laplace(Grayscale_Image)

Image_Sharpenned_auto=Grayscale_Image-Image_Lapplaced_auto
Image_Sharpenned_auto=tools.float_to_uint8(Image_Sharpenned_auto)

#-------------------------------------------------------------------------------
mask_manual=np.ones((3,3))
mask_manual[1,1]=-8

rows=len(Grayscale_Image)
columns=len(Grayscale_Image[0])

Image_Lapplaced_manual=np.zeros((rows,columns))

Image_Lapplaced_manual=scipy.ndimage.convolve(Grayscale_Image,mask_manual)
Image_Lapplaced_manual=tools.float_to_uint8(Image_Lapplaced_manual)

#-------------------------------------------------------------------------------
Image_Sharpenned_manual=Grayscale_Image-Image_Lapplaced_manual
Image_Sharpenned_manual=tools.float_to_uint8(Image_Sharpenned_manual)

```











Sharpening with Unsharp method

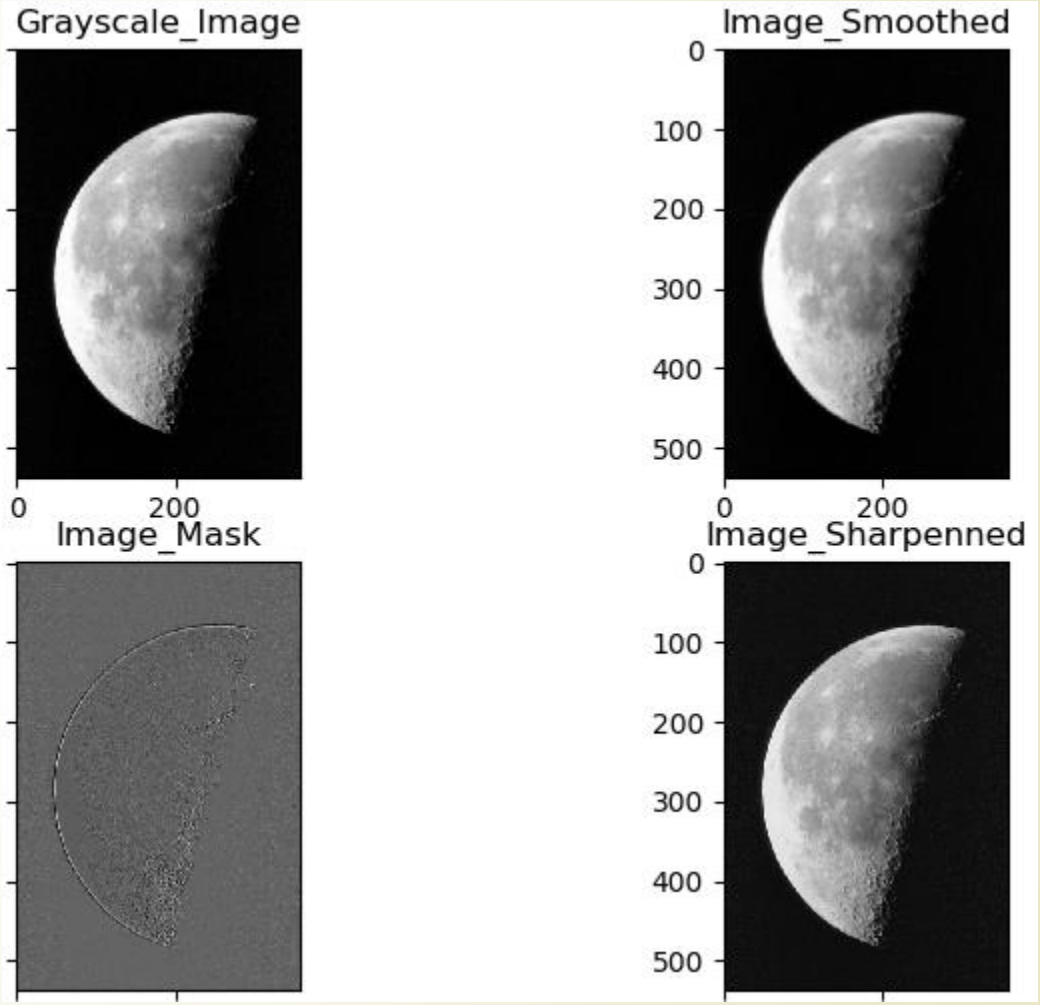
- Είναι μια διαδικασία που έχει χρησιμοποιηθεί για πολλά χρόνια από τη βιομηχανία εκτύπωσης και εκδόσεων για να οξύνει εικόνες. Αποτελείται από την αφαίρεση μιας εξομαλυμένης έκδοσης μιας εικόνας από την αρχική.
- Η διαδικασία ονομάζεται unsharpmasking,και αποτελείτε από τα παρακάτω βήματα:

1. Θόλωμα της αρχικής εικόνας: $f(x,y)$–> $\bar{f}(x,y)$
2. Αφαίρεση της θολωμένης από την αρχική για να πάρουμε τη μάσκα: $gmask(x,y)=f(x,y)-\bar{f}(x,y)$
1. Πρόσθεση της μάσκας στην αρχική: $g(x,y)=f(x,y)+k\cdot gmask(x,y)$

k=1 ,unsharp filtering
K>1, high boost filtering

Sharpening with Unsharp method


```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import Image_formats_and_conversions as tools
import scipy.ndimage


Grayscale_Image = mpimg.imread('moon.tiff')

Grayscale_Image= np.asarray(Grayscale_Image, dtype = np.float64 )
#-----------------------------------------------------------------
mask_mean=np.ones((3,3))/9

Image_Smoothed=scipy.ndimage.convolve(Grayscale_Image,mask_mean)
Image_Smoothed=tools.float_to_uint8(Image_Smoothed)

Image_Mask=Grayscale_Image-Image_Smoothed

Image_Sharpenned=Grayscale_Image+Image_Mask

#-----------------------------------------------------------------


figure1=plt.figure(1)

subplot1=figure1.add_subplot(2,2,1)
plt.imshow(Grayscale_Image,cmap="gray")
subplot1.set_title('Grayscale_Image')

subplot1=figure1.add_subplot(2,2,2)
plt.imshow(Image_Smoothed,cmap="gray")
subplot1.set_title('Image_Smoothed')

subplot1=figure1.add_subplot(2,2,3)
plt.imshow(Image_Mask,cmap="gray")
subplot1.set_title('Image_Mask')

subplot1=figure1.add_subplot(2,2,4)
plt.imshow(Image_Sharpenned,cmap="gray")
subplot1.set_title('Image_Sharpenned')

```



Η πρώτη παράγωγος για μη γραμμική όξυνση εικόνας (Gradient-κλίση εικόνας)

Οι πρώτοι παράγωγοι στην επεξεργασία εικόνας μπορούν να υλοποιηθούν με χρήση του πλάτους του διανύσματος της κλίσης!
Για μια εικόνα$f(x,y)$, η κλίσης της f στις συντεταγμένες (x,y) ορίζεται ως ο πίνακας στήλη:

$$\nabla f = grad(f) = \begin{pmatrix} g_x \\ g_y \end{pmatrix} = \begin{bmatrix} \frac{\vartheta f(x,y)}{\vartheta x} \\ \frac{\vartheta f(x,y)}{\vartheta y} \end{bmatrix}$$

Το πλάτος του δ$ιανύσματος\nabla f$είναι η εικόνα κλίσης:

$$M(x,y)=mag(\nabla f)=\sqrt{g_x^2 + g_y^2} \approx |g_x| + |g_y|$$

Η πρώτη παράγωγος για μη γραμμική όξυνση εικόνας (Gradient-κλίση εικόνας)x,y

f(x - 1, y - 1)	f(x - 1, y)	f(x - 1, y + 1)
f(x, y - 1)	f(x,y)	f(x, y + 1)
f(x + 1, y - 1)	f(x + 1, y)	f(x + 1, y + 1)

→

-1	-2	-1
0	0	0
1	2	1

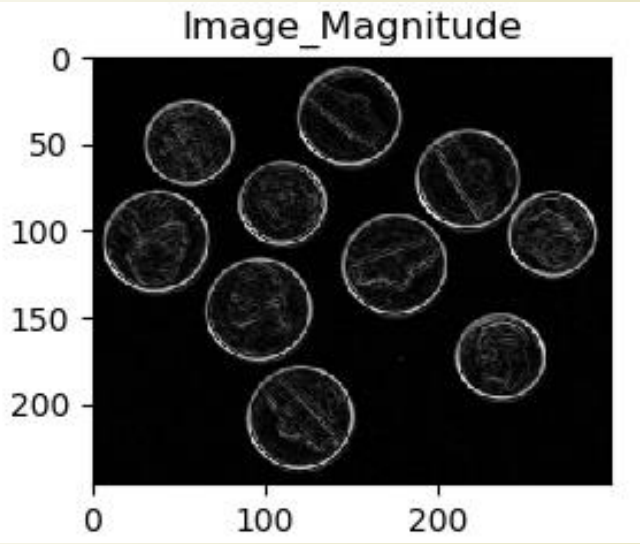
f(x - 1, y - 1)	f(x - 1, y)	f(x - 1, y + 1)
f(x, y - 1)	f(x,y)	f(x, y + 1)
f(x + 1, y - 1)	f(x + 1, y)	f(x + 1, y + 1)

→

-1	0	1
-2	0	2
-1	0	1

Η πρώτη παράγωγος για μη γραμμική όξυνση εικόνας (Gradient-κλίση εικόνας)x,y,Auto

```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import scipy.ndimage

Grayscale_Image = mpimg.imread('coins.png')

Grayscale_Image= np.asarray(Grayscale_Image, dtype = np.float64 )

#-----------------------------------------------------------------------------
# Compute the Sobel filter for two values of directions

Image_Gradient_x = scipy.ndimage.sobel(Grayscale_Image, 0)  # horizontal derivative
Image_Gradient_y = scipy.ndimage.sobel(Grayscale_Image, 1)  # vertical derivative
Image_Magnitude = np.hypot(Image_Gradient_x, Image_Gradient_y)  # magnitude
Image_Magnitude *= 255.0 / np.max(Image_Magnitude)  # normalize (Q&D)
#-----------------------------------------------------------------------------

figure1=plt.figure(1)

subplot1=figure1.add_subplot(1,3,1)
plt.imshow(Grayscale_Image,cmap="gray")
subplot1.set_title('Grayscale_Image')

subplot1=figure1.add_subplot(1,2,2)
plt.imshow(Image_Magnitude,cmap="gray")
subplot1.set_title('Image_Magnitude')

```



Η πρώτη παράγωγος για μη γραμμική όξυνση εικόνας (Gradient-κλίση εικόνας)x,y,Manual

```python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import scipy.ndimage

Grayscale_Image = mpimg.imread('coins.png')

Grayscale_Image= np.asarray(Grayscale_Image, dtype = np.float64 )

#-------------------------------------------------------------------------------
Gradient_Mask_x=[[-1,-2,-1],[0,0,0],[1,2,1]]
Gradient_Mask_x= np.asarray(Gradient_Mask_x, dtype = np.int32 )

Gradient_Mask_y=np.transpose(Gradient_Mask_x)

Image_Gradient_x=scipy.ndimage.convolve(Grayscale_Image,Gradient_Mask_x)


Image_Gradient_y=scipy.ndimage.convolve(Grayscale_Image,Gradient_Mask_y)

#-------------------------------------------------------------------------------
Image_Gradient_x_y=Image_Gradient_x+Image_Gradient_y

Image_Gradient_x_y_power_2=np.power(Image_Gradient_x,2)+np.power(Image_Gradient_y,2)

Image_Gradient_x_y_square_root=np.sqrt(Image_Gradient_x_y_power_2)
Image_Gradient_x_y_square_root= np.asarray(Image_Gradient_x_y_square_root, dtype = np.float64 )

Image_Gradient_x_y_abs=np.abs(Image_Gradient_x)+np.abs(Image_Gradient_y)

#-------------------------------------------------------------------------------
```