Image Gradients

- Sobel filter
- Scharr filter
- Laplacian Derivatives

Review of all filter

In [2]:

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('images/sudoku.png',0)
# works only with grayscale images

blur = cv2.blur(img,(5,5))


sobelx = cv2.Sobel(blur,-1,1,0,ksize=5)
# ddepth - output datatype (-1 same as input)
# x - order of derivative along x (can be 0 or 1)
# y - order of derivative along y (can be 0 or 1)
# ksize - kernal size (-1 mean scharr derivative)
# delta - delta is added to the output image


sobely = cv2.Sobel(blur,-1,0,1,ksize=5)
laplacian = cv2.Laplacian(img,cv2.CV_64F,ksize = 5,delta = 120)


abs_dst = cv2.convertScaleAbs(laplacian)

cv2.imshow('img',img)

cv2.imshow('sobelx',sobelx)
cv2.imshow('sobely',sobely)

cv2.imshow('laplacian',laplacian)


cv2.waitKey(0)
cv2.destroyAllWindows()
```

Sobel Filter

In [10]:

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

```
src = cv2.imread('images/box.jpg',0)

grad_x = cv2.Sobel(scr, -1, 1, 0,scale = 1, delta=delta)
# Gradient-Y
# grad_y = cv.Scharr(gray,ddepth,0,1)
# grad_y = cv2.Sobel(gray, ddepth, 0, 1, ksize=3, scale=scale, delta=delta, borderType=


# abs_grad_x = cv2.convertScaleAbs(grad_x)
# abs_grad_y = cv2.convertScaleAbs(grad_y)


# grad = cv2.addWeighted(abs_grad_x, 0.5, abs_grad_y, 0.5, 0)


# cv2.imshow("grad", grad)
cv2.imshow("grad", grad_x)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

In [2]:

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('images/box.jpg',0)

# Output dtype = cv2.CV_8U
sobelx8u = cv2.Sobel(img,cv2.CV_8U,1,0,ksize=-1)

# Output dtype = cv2.CV_64F. Then take its absolute and convert to cv2.CV_8U
sobelx64f = cv2.Sobel(img,cv2.CV_64F,1,0,ksize=5)
abs_sobel64f = np.absolute(sobelx64f)
sobel_8u = np.uint8(abs_sobel64f)


cv2.imshow('sobel_8u',sobel_8u)
cv2.imshow('sobelx8u',sobelx8u)


cv2.waitKey(0)
cv2.destroyAllWindows()

# plt.subplot(1,3,1),plt.imshow(img,cmap = 'gray')
# plt.title('Original'), plt.xticks([]), plt.yticks([])
# plt.subplot(1,3,2),plt.imshow(sobelx8u,cmap = 'gray')
# plt.title('Sobel CV_8U'), plt.xticks([]), plt.yticks([])
# plt.subplot(1,3,3),plt.imshow(sobel_8u,cmap = 'gray')
# plt.title('Sobel abs(CV_64F)'), plt.xticks([]), plt.yticks([])

# plt.show()
```

sobel

In [3]:

```
from cv2 import cv2
```

```
import numpy as np

img = cv2.imread('images/box.jpg',0)

sobelx = cv2.Sobel(img,cv2.CV_8U,1,0,ksize = 3,scale = 1,
                   delta = 0)
sobely = cv2.Sobel(img,cv2.CV_8U,0,1,ksize = 3,scale = 1,
                   delta = 0)

sobelx64f = cv2.Sobel(img,cv2.CV_64F,1,0,ksize=3)

abs_sobel64f = np.absolute(sobelx64f)
sobel_8u = np.uint8(abs_sobel64f)


cv2.imshow('img',img)

cv2.imshow('x',sobelx)
cv2.imshow('y',sobely)
cv2.imshow('sobel_8u',sobel_8u)


cv2.waitKey(0)
cv2.destroyAllWindows()
```

We can force sobel function to act as scharr fucntion by providing a ksize of -1

Scharr function uses a bit modified kernal to remove the noices which is otherwise there in sobel function

numpy visualize

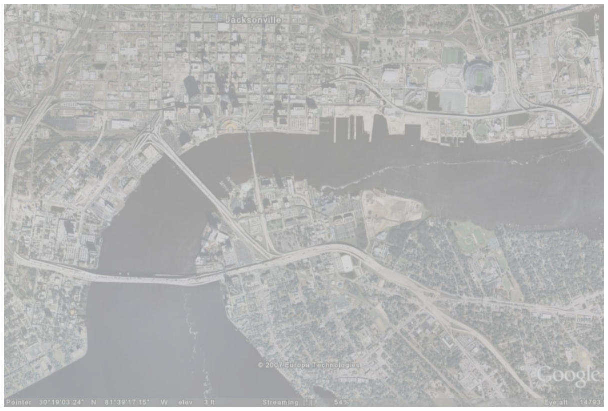
In [4]:

```
from cv2 import cv2
import numpy as np

img = np.zeros((7,7))
# img = np.zeros((7,7),dtype = 'uint8')

img[2:5,2:5] = 1


print(img)

sobelx = cv2.Sobel(img,cv2.CV_16S,1,0,ksize = 3,scale = 1, delta = 0, borderType= cv2.B
# sobely = cv2.Sobel(img,cv2.CV_8U,0,1,ksize = 3,scale = 1, delta = 0, borderType= cv2.

# sobelx64f = cv2.Sobel(img,cv2.CV_64F,1,0,ksize=3)

abs_sobel64f = np.absolute(sobelx)
sobel_8u = np.uint8(abs_sobel64f)

print(sobel_8u)

# cv2.imshow('img',img)
# cv2.imshow('x',sobelx)
# cv2.imshow('y',sobely)
# cv2.imshow('sobel_8u',sobel_8u)


# cv2.waitKey(0)
# cv2.destroyAllWindows()
```

```
[[0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 1. 1. 0. 0.]
 [0. 0. 1. 1. 1. 0. 0.]
 [0. 0. 1. 1. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]]
[[0 0 0 0 0 0 0]
 [0 1 1 0 1 1 0]
 [0 3 3 0 3 3 0]
 [0 4 4 0 4 4 0]
 [0 3 3 0 3 3 0]
 [0 1 1 0 1 1 0]
 [0 0 0 0 0 0 0]]
```

Home Work

1) Do histogram equalisation on the below image



In [ ]:

```
```