

```

1 import cv2 as cv
2 import matplotlib.pyplot as plt
3 import numpy as np
4 import math
5
6 # path to the input img
7 path = 'C:/Users/Raiyan/Desktop/building.jpg'
8
9 # reading img + converting from BGR to GRAY
10 img = cv.imread(path)
11 img = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
12
13 k_h = int(input("Enter kernel height: "))
14 k_w = k_h
15 k_size = (k_h, k_w)
16
17 # kernel with neg center value
18 kernel = np.array([[0,-1,0],
19                    [-1,+4,-1],
20                    [0,-1,0]], np.float32)
21
22 # img height
23 img_h = img.shape[0]
24 # img width
25 img_w = img.shape[1]
26 # kernel height // 2
27 a = kernel.shape[0] // 2
28 # kernel width // 2
29 b = kernel.shape[1] // 2
30
31 # empty op img
32 output = np.zeros((img_h, img_w), np.float32)
33
34 # conv
35 # visiting each pixel in the img
36 # m ta row img e ... for each row ...
37 for i in range(img_h):
38     # n ta coln img e ... for each coln ...
39     for j in range(img_w):
40         # empty var for storing all the values
41         values = []
42         # visiting each pixel in the kernel
43         # a ta row img e ... for each row ...
44         for x in range(-a, a+1):
45             # b ta coln img e ... for each coln ...
46             for y in range(-b, b+1):
47                 if 0 <= i-x < img_h and 0 <= j-y < img_w:
48                     output[i][j] += kernel[a+x][b+y] * img[i-x][j-y]
49                 else:
50                     output[i][j] += 0
51
52 out_conv = output
53
54 # scaled
55 def scaled(image):
56     g_m = image - image.min()
57     g_s = 255*(g_m / g_m.max())
58     return g_s.astype(np.float32)
59
60 scaled = scaled(out_conv)
61
62 plt.imshow(scaled, 'gray')
63 plt.title('scaled')
64 plt.show()
65
66 output = out_conv
67
68 # val capping or clipping from 0 - 255
69 for i in range(img_h):
70     for j in range(img_w):
71         if output[i][j] < 0 :
72             output[i][j] = 0
73         elif output[i][j] > 255 :
74             output[i][j] = 255
75
76 clipped = output.astype(np.float32)
77

```

```

78 # center of kernel is (+)
79 sharpened = img + out_conv
80
81 output = sharpened
82 # sharpened + clipping from 0 - 255
83 for i in range(img_h):
84     for j in range(img_w):
85         if output[i][j] < 0 :
86             output[i][j] = 0
87         elif output[i][j] > 255 :
88             output[i][j] = 255
89
90 sharpened_and_clipped = output.astype(np.float32)
91
92 # sharpened + scaled
93 g_m = sharpened - sharpened.min()
94 g_s = (g_m / g_m.max()) * 255
95 sharpened_and_scaled = g_s.astype(np.float32)
96
97
98 def show_images(images, image_title):
99     # displaying multiple images side by side
100     # https://stackoverflow.com/questions/41793931/plotting-images-side-by-side-
    using-matplotlib
101
102     # err : was giving weird colormap due to diff in the mechanism of reading img of
    cv2 & matplotlib
103     # https://stackoverflow.com/questions/3823752/display-image-as-grayscale-using-
    matplotlib
104     # running this once in the code will ALWAYS give gray op
105     plt.gray()
106
107     no_of_imgs = len(images)
108     f = plt.figure()
109     for i in range(no_of_imgs):
110
111         # Debug, plot figure
112         axes = f.add_subplot(1, no_of_imgs, i + 1)
113         # the last img will show y axis on the RHS instead of LHS(which is by
    default)
114
115         if i==no_of_imgs-1:
116             axes.yaxis.tick_right()
117
118         plt.title(image_title[i])
119         plt.imshow(images[i])
120         # plt.rc('font', size=8)
121         plt.show(block=True)
122
123
124 show_images([img,scaled], ['input', 'scaled'])
125 show_images([clipped, sharpened_and_scaled], ['clipped', 'sharp + scaled'])
126 show_images([img,sharpened_and_scaled], ['input', 'final output'])
127
128
129
130

```



