

```

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 # avg kernel
18 kernel = np.zeros( k_size, np.float32)
19
20 # img height
21 img_h = img.shape[0]
22 # img width
23 img_w = img.shape[1]
24 # kernel height // 2
25 a = kernel.shape[0] // 2
26 # kernel width // 2
27 b = kernel.shape[1] // 2
28
29 pi=3.1416
30 sigma = 1.0
31 normalizing_c = - 1.0 / ( sigma * sigma * sigma * sigma * pi )
32
33 # building kernel1
34 for x in range(-a,a+1):
35     for y in range(-b,b+1):
36         val1 = math.exp( -(x*x + y*y) / (2.0 * sigma * sigma) )
37         val2 = 1 - ((x*x + y*y) / (2.0 * sigma * sigma))
38         val1 = val1 * val2 * normalizing_c
39         kernel[a+x][b+y] = val1
40
41 # empty op img
42 output = np.zeros((img_h,img_w), np.float32)
43
44 # conv
45 # visiting each pixel in the img
46 # m ta row img e ... for each row ...
47 for i in range(img_h):
48     # n ta coln img e ... for each coln ...
49     for j in range(img_w):
50         # visiting each pixel in the kernel
51         # a ta row img e ... for each row ...
52         for x in range(-a,a+1):
53             # b ta coln img e ... for each coln ...
54             for y in range(-b,b+1):
55                 if 0 <= i-x < img_h and 0 <= j-y < img_w:
56                     output[i][j] += kernel[a+x][b+y] * img[i-x][j-y]
57                 else:
58                     output[i][j] += 0
59
60 out_conv = output
61 # scaled
62 def scaled(image):
63     g_m = image - image.min()
64     g_s = 255*(g_m / g_m.max())
65     return g_s.astype(np.float32)
66
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 output = output.astype(np.float32)
77

```

```
78 clipped = output
79 scaled_output = scaled(out_conv)
80 img1 = img
81 #####
82 img = img.astype(np.float32)
83 img -= clipped
84
85 # val capping or clipping from 0 - 255
86 for i in range(img_h):
87     for j in range(img_w):
88         if img[i][j] < 0 :
89             img[i][j] = 0
90         elif img[i][j] > 255 :
91             img[i][j] = 255
92
93 def show_images(images, image_title):
94     # displaying multiple images side by side
95     # https://stackoverflow.com/questions/41793931/plotting-images-side-by-side-
96     # using-matplotlib
97     # err : was giving weird colormap due to diff in the mechanism of reading img of
98     # cv2 & matplotlib
99     # https://stackoverflow.com/questions/3823752/display-image-as-grayscale-using-
100     # matplotlib
101     # running this once in the code will ALWAYS give gray op
102     plt.gray()
103
104     no_of_imgs = len(images)
105     f = plt.figure()
106     for i in range(no_of_imgs):
107         # Debug, plot figure
108         axes = f.add_subplot(1, no_of_imgs, i + 1)
109         # the last img will show y axis on the RHS instead of LHS(which is by
110         # default)
111         if i==no_of_imgs-1:
112             axes.yaxis.tick_right()
113
114         plt.title(image_title[i])
115         plt.imshow(images[i], 'gray')
116         # plt.rc('font', size=8)
117     plt.show(block=True)
118
119 show_images([img1,img], ['input', 'output'])
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
```