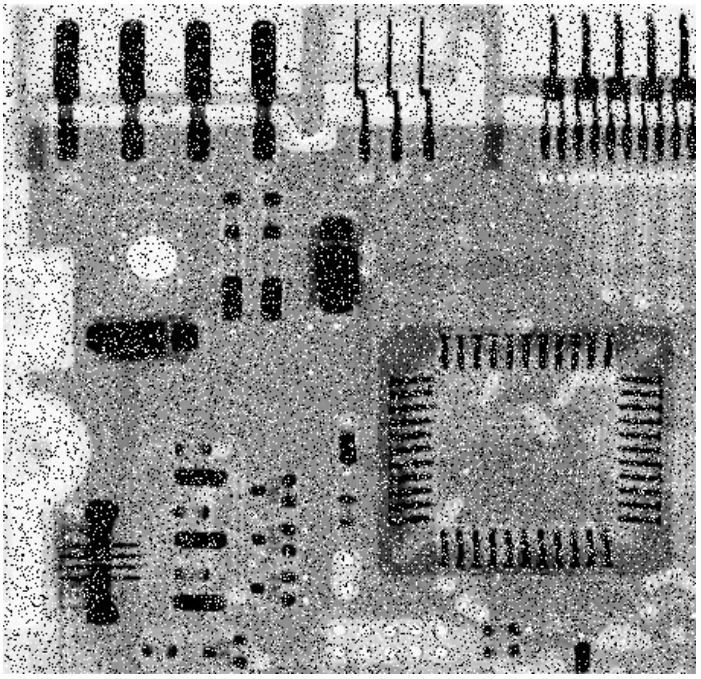
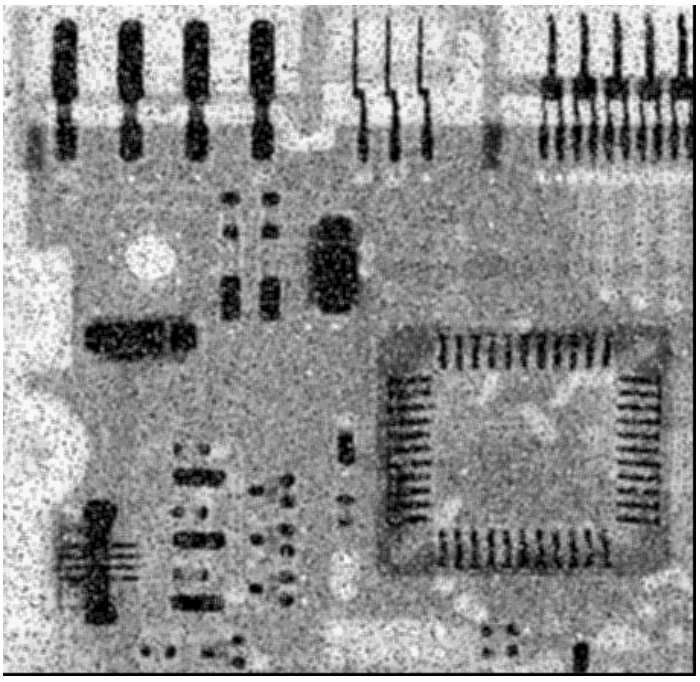
```
# Low Pass SPatial Domain Filtering
# to observe the blurring effect


import cv2
import numpy as np
from google.colab.patches import cv2_imshow # for image display


# Read the image
#img = cv2.imread('sample.png', 0)
img = cv2.imread("/content/drive/My Drive/colab/noisysalterpepper.png", 0)
cv2_imshow(img)
```



```
# Obtain number of rows and columns
# of the image
m, n = img.shape

# Develop Averaging filter(3, 3) mask
mask = np.ones([3, 3], dtype = int)
mask = mask / 9

#k_m, k_n = mask.shape

# Convolve the 3X3 mask over the image
img_new = np.zeros([m, n])
#mask.shape
```

```
(3, 3)
```

```
for i in range(1, m-1):
    for j in range(1, n-1):
        temp = img[i-1, j-1]*mask[0, 0]+img[i-1, j]*mask[0, 1]+
        img[i-1, j + 1]*mask[0, 2]+img[i, j-1]*mask[1, 0]+
        img[i, j]*mask[1, 1]+img[i, j + 1]*mask[1, 2]+
        img[i + 1, j-1]*mask[2, 0]+img[i + 1, j]*mask[2, 1]+
        img[i + 1, j + 1]*mask[2, 2]
        img_new[i-1, j-1]= temp

img_new = img_new.astype(np.uint8)
cv2_imshow(img_new)
```



```
# Obtain the number of rows and columns
# of the image
m, n = img.shape

# Traverse the image. For every 3X3 area,
# find the median of the pixels and
# replace the center pixel by the median
img_new1 = np.zeros([m, n])

for i in range(1, m-1):
    for j in range(1, n-1):
        temp = [img[i-1, j-1],
               img[i-1, j],
               img[i-1, j + 1],
               img[i, j-1],
               img[i, j],
               img[i, j + 1],
               img[i + 1, j-1],
               img[i + 1, j],
               img[i + 1, j + 1]]

        temp = sorted(temp)
        img_new1[i, j]= temp[4]

img_new1 = img_new1.astype(np.uint8)
cv2_imshow(img_new1)


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

```
array([[0.04, 0.04, 0.04, 0.04, 0.04],
       [0.04, 0.04, 0.04, 0.04, 0.04],
       [0.04, 0.04, 0.04, 0.04, 0.04],
       [0.04, 0.04, 0.04, 0.04, 0.04],
       [0.04, 0.04, 0.04, 0.04, 0.04]], dtype=float32)
```

```
dst = cv2.filter2D(img,-1,kernel)
cv2_imshow(dst)




gauss_x = cv2.getGaussianKernel(5, 1);
gauss_y = cv2.getGaussianKernel(5, 1);
gauss = gauss_x * gauss_y.transpose()
```

```
dst = cv2.filter2D(img,-1,gauss)
cv2_imshow(dst)
```

✓ 6s completed at 2:34 PM