Image Processing | Lab 6

Different types of Noise

Raj Panchal
17CEUBG104

Task 1 : Synthesize the image of a chess board. (use intensity 50 for dark block and 170 for bright block). Add gamma noise and exponential noise (both separately) and generate noisy images. Show and comment on histogram of the noisy images.

Code:

Function for gamma noise:

```
function [noisy, y] = gamma_noise(img,a,b)
  [M,N]=size(img);
  n = zeros(M,N);
  k=-1/a;
  for i=1:b,
    n=n+k*log(1-rand(M,N));
  endfor
  noisy=n+img;
  y=imhist(noisy);
endfunction
```

Function for exponential noise:

```
function [noisy, y] = exponential_noise(img,a,b)
  [M,N]=size(img);
  n = zeros(M,N);
  k=-1/a;
  n=k*log(1-rand(M,N));
  noisy=n+img;
  y=imhist(noisy);
endfunction
```

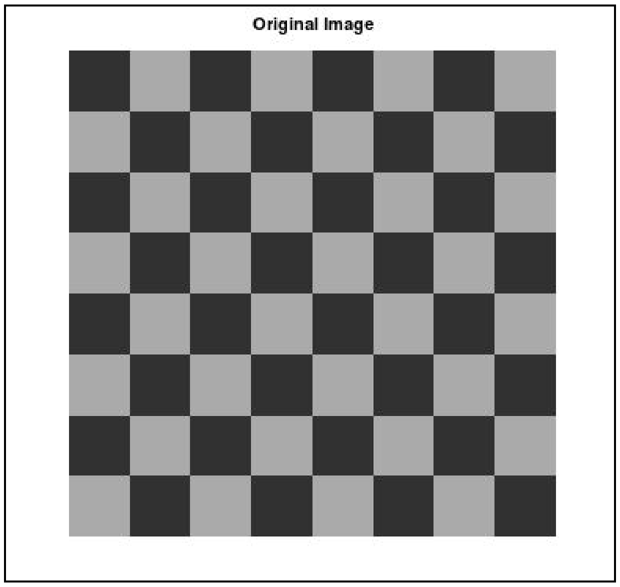
Main :

```
r = (checkerboard > 0.5);
[m,n] = size(r);
chess_image = uint8(zeros(size(r)));
for i=1:m,
  for j=1:n,
    if r(i,j)==1
      chess_image(i,j)=170;
     else
      chess_image(i,j)=50;
     endif
   endfor
endfor

imshow(chess_image);
```

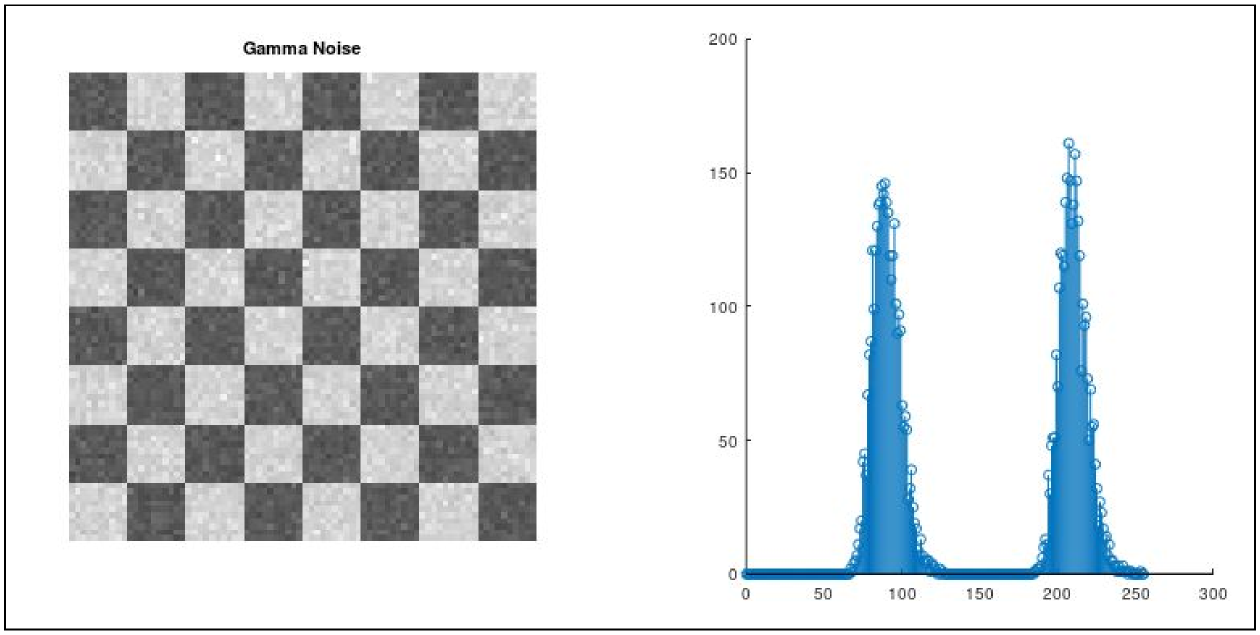
```
figure;

[noisy_image, y] = gamma_noise(chess_image,0.5,20);
subplot(1,2,1);
imshow(noisy_image);
subplot(1,2,2);
stem(0:255,y);

figure;

[noisy_image, y] = exponential_noise(chess_image,0.5,20);
subplot(1,2,1);
imshow(noisy_image);
subplot(1,2,2);
stem(0:255,y);
```

Output :

Original Image:

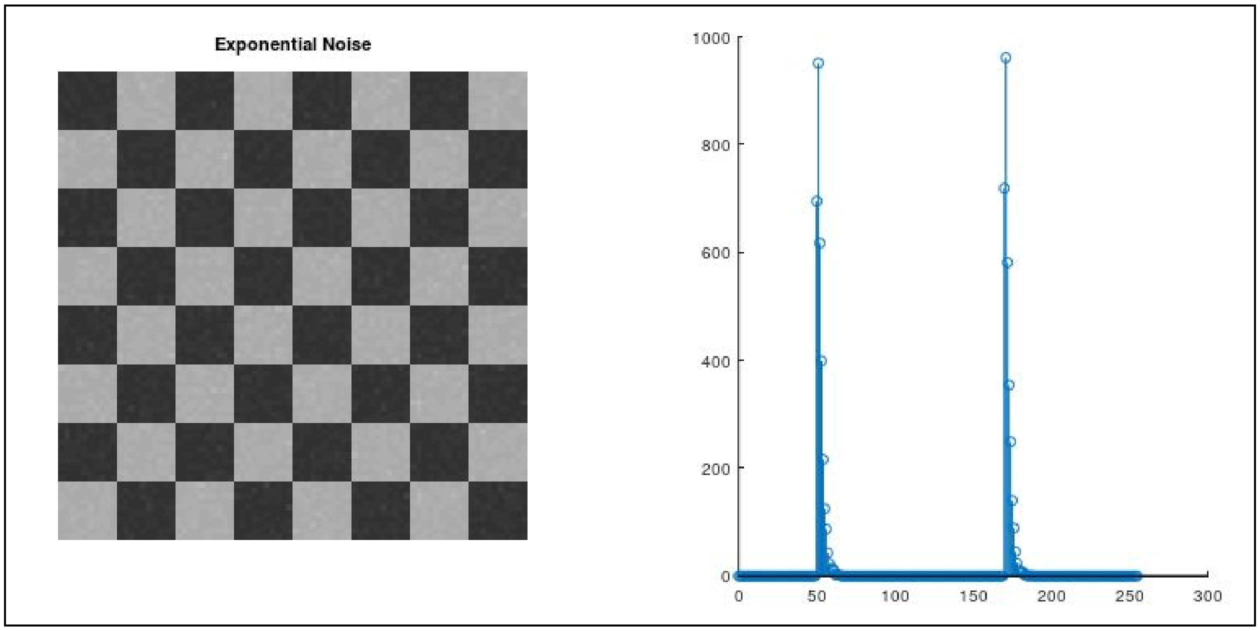


Gamma noise :



- The histogram is the same as given in the book for gamma noise.

Exponential noise :

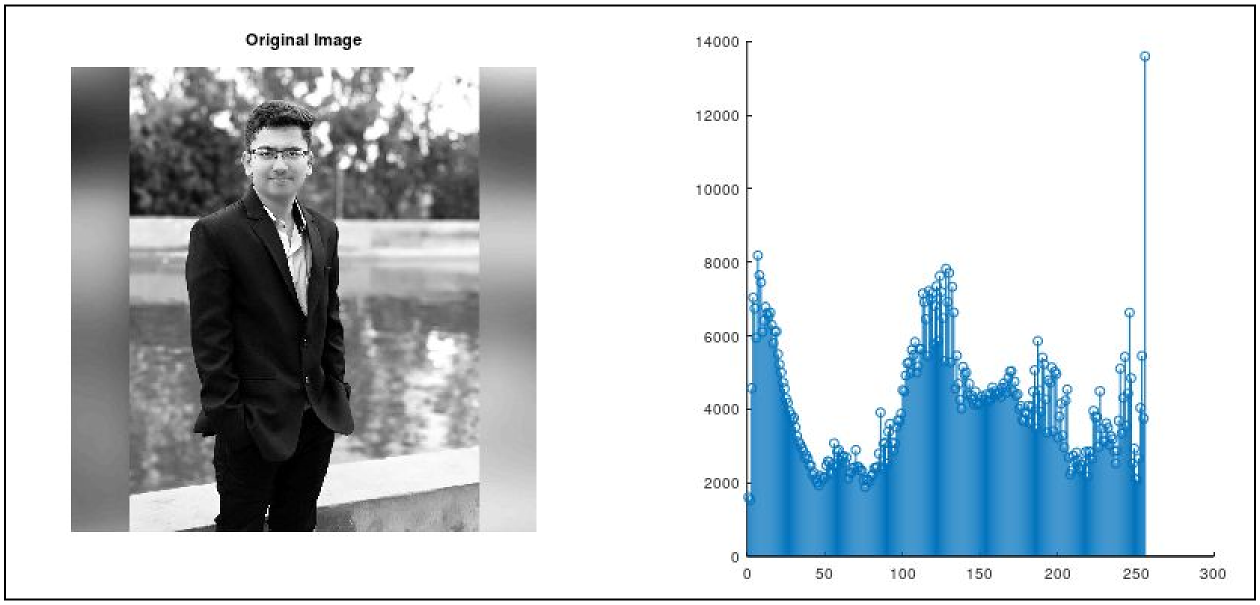


- The histogram is the same as given in the book for exponential noise.

Task 2 : Take your grayscale photo and generate noisy photo help of given information:

Original Image :

```
r = imread("gray_scale_img.jpg");
subplot(1,2,1);
imshow(r);
title("Original Image");
x = imhist(r);
subplot(1,2,2);
stem(x);
```



1. Gaussian noise with probability 0.4 using randn function.

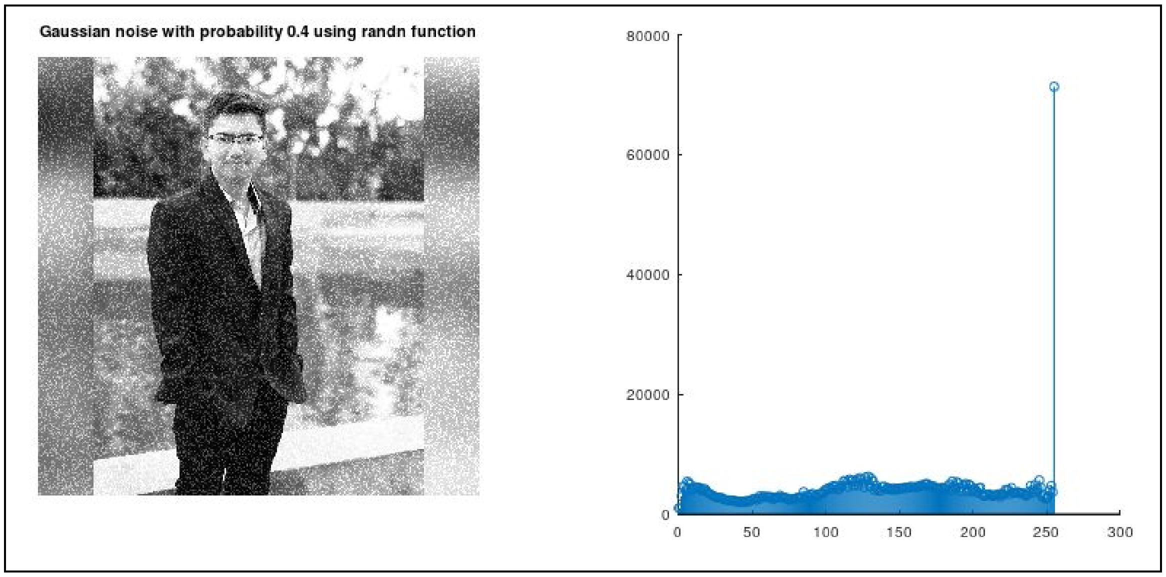
Function for gaussian noise :

```
function [noisy,y] = gaussian_with_prob(img,mu,sigma,prob)
  [M,N] = size(img);
   n = mu + (round(sigma*randn(M,N)));
   n = uint8(n);
   noisy=img;
  for i=1:M*N*prob,
    p=round(1+(M-1)*rand());
    q=round(1+(N-1)*rand());
    noisy(p,q) = img(p,q)+n(p,q);
  endfor
  y = imhist(noisy);
  noisy=uint8(noisy);
endfunction
```

Main code:

```
[noisy,y] = gaussian_with_prob(r,50,20,0.4);
subplot(1,2,1);
imshow(noisy);
title("Gaussian noise with probability 0.4 using randn function")
subplot(1,2,2);
stem(0:255,y);
imwrite(noisy,"Gaussian_with_0.4.jpg");
```

Output:

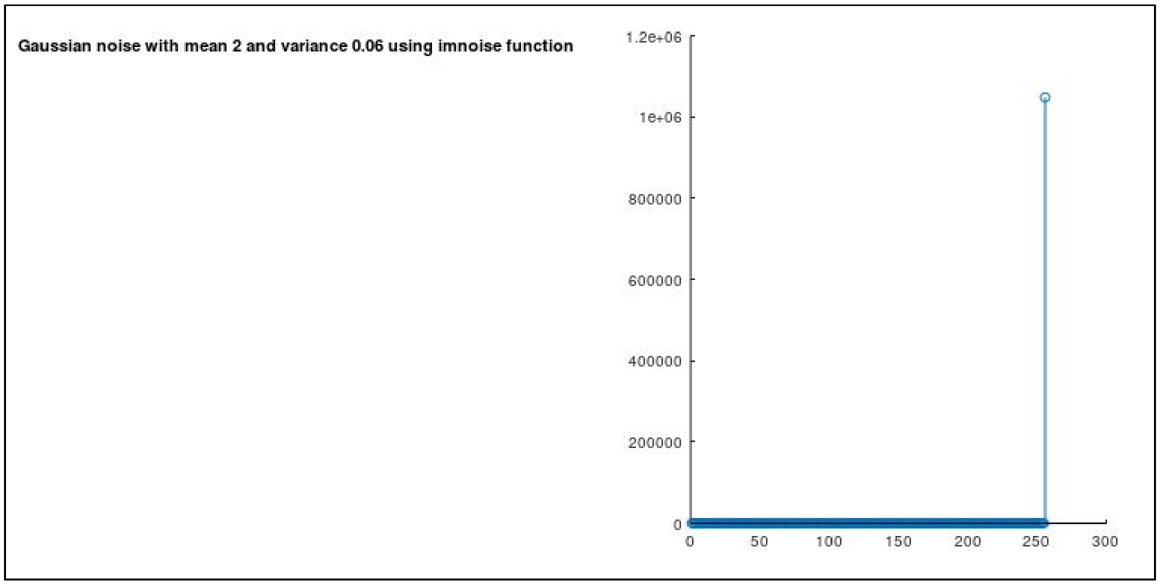


2. Gaussian noise with mean 2 and variance 0.06 using imnoise function.

Code:

```
s = imnoise(r,'gaussian',2,0.06);
subplot(1,2,1);
imshow(s);
title("Gaussian noise with mean 2 and variance 0.06 using imnoise function");
x = imhist(s);
subplot(1,2,2);
stem(x);
imwrite(s,"Gaussian_m2_v0.06.jpg");
```

Output:



3. **Salt and pepper noise with probability 0.2 using your user defined function.**

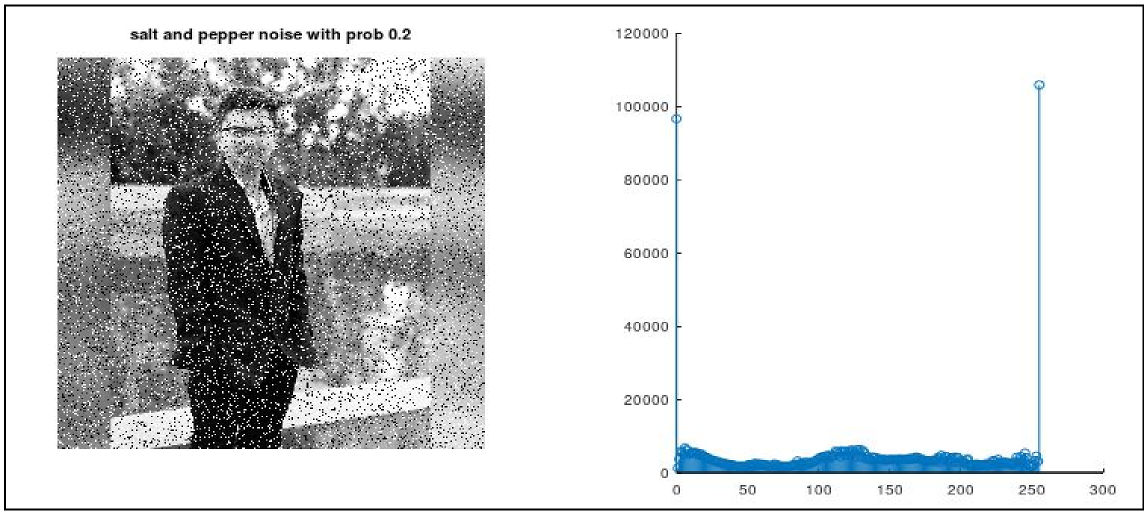
Function for salt pepper noise :

```
function [noisy,y] = salt_and_pepper(img,prob)
  [M,N] = size(img);
  noisy = img;
  for i=1:M*N*prob,
    p=round(1+(M-1)*rand());
    q=round(1+(N-1)*rand());
    if (round(rand())==0)
      img(p,q)=0;
    else
      img(p,q)=255;
    end
  endfor
  noisy = img;
  y = imhist(noisy);
endfunction
```

Main code:

```
[noisy_im, y] = salt_and_pepper(r,0.2);
subplot(1,2,1);
imshow(noisy_im);
title("salt and pepper noise with prob 0.2");
subplot(1,2,2);
stem(0:255,y);
imwrite(noisy_im,"salt_pepper_with_0.2.jpg");
```

Output:



4. Salt noise with probability 0.5 using your user defined function.

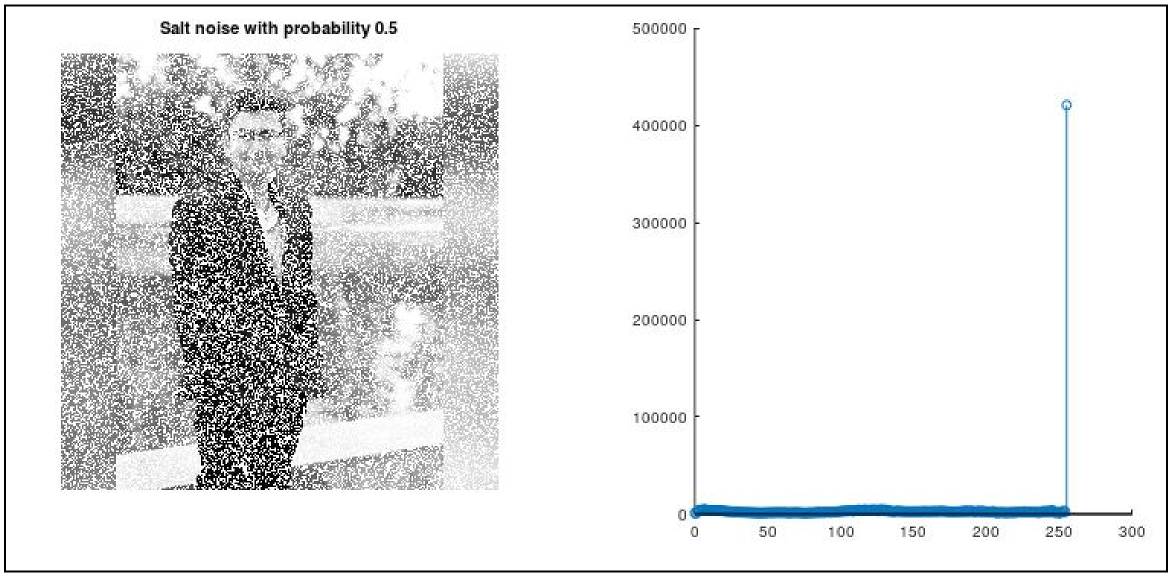
Function for salt noise :

```
function [noisy,y] = salt_noise(img,prob)
  [M,N] = size(img);
  noisy = img;
  for i=1:M*N*prob,
    p=round(1+(M-1)*rand());
    q=round(1+(N-1)*rand());
    img(p,q)=255;
  endfor
  noisy = img;
  y = imhist(noisy);
endfunction
```

Main code :

```
[noisy_im, y] = salt_noise(r,0.5);
subplot(1,2,1);
imshow(noisy_im);
title("Salt noise with probability 0.5");
subplot(1,2,2);
stem(0:255,y);
imwrite(noisy_im,"salt_with_0.5p.jpg");
```

Output :



5. Pepper noise with probability 0.3 using your user defined function.

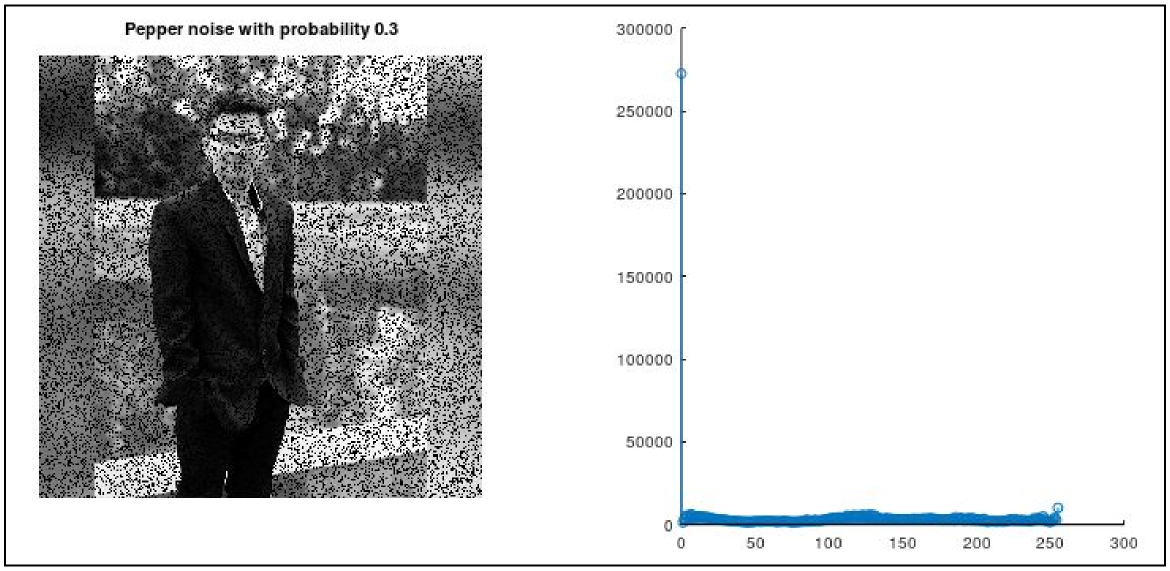
Function for pepper noise :

```
function [noisy,y] = pepper_noise(img,prob)
  [M,N] = size(img);
  noisy = image;
  for i=1:M*N*prob,
    p=round(1+(M-1)*rand());
    q=round(1+(N-1)*rand());
    img(p,q)=0;
  endfor
  noisy = img;
  y = imhist(noisy);
endfunction
```

Main code :

```
[noisy_im, y] = pepper_noise(r,0.3);
subplot(1,2,1);
imshow(noisy_im);
title("Pepper noise with probability 0.3");
subplot(1,2,2);
stem(0:255,y);
imwrite(noisy_im,"pepper_with_0.3p.jpg");
```

Output :



6. Uniform noise with probability 1 and a=10 , b=20.

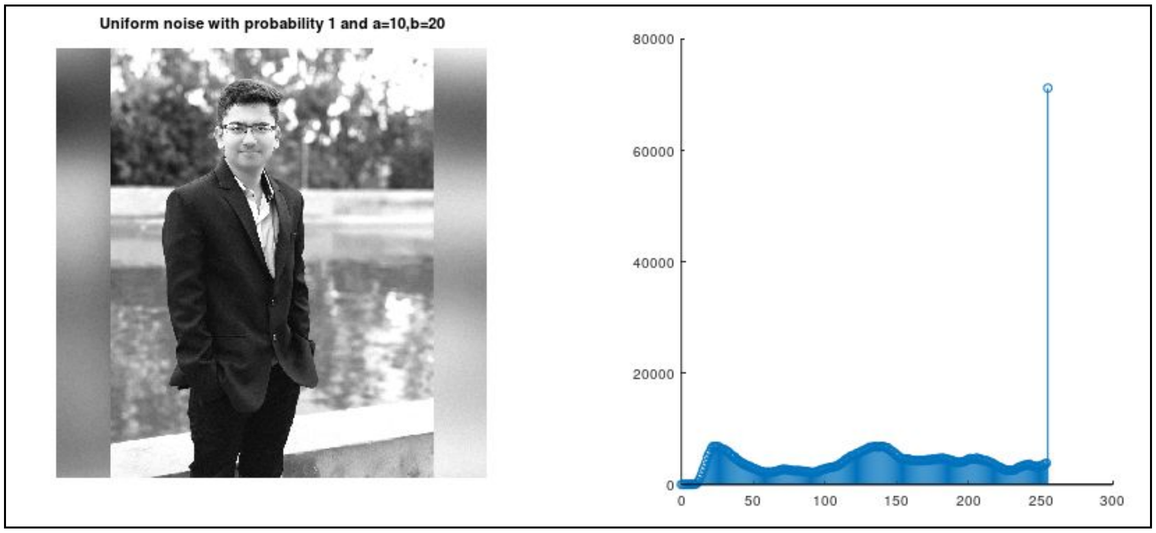
Function for uniform noise :

```
function [noisy,y] = uniform_noise(img,a,b)
  [M,N] = size(img);
  n = uint8(round(a+(b-a)*rand(M,N)));
  noisy = img+n;
  y = imhist(noisy);
endfunction
```

Main code :

```
[noisy_im, y] = uniform_noise(r,10,20);
subplot(1,2,1);
imshow(noisy_im);
title("Uniform noise with probability 1 and a=10,b=20");
subplot(1,2,2);
stem(0:255,y);
imwrite(noisy_im,"uniform_noise.jpg");
```

Output:

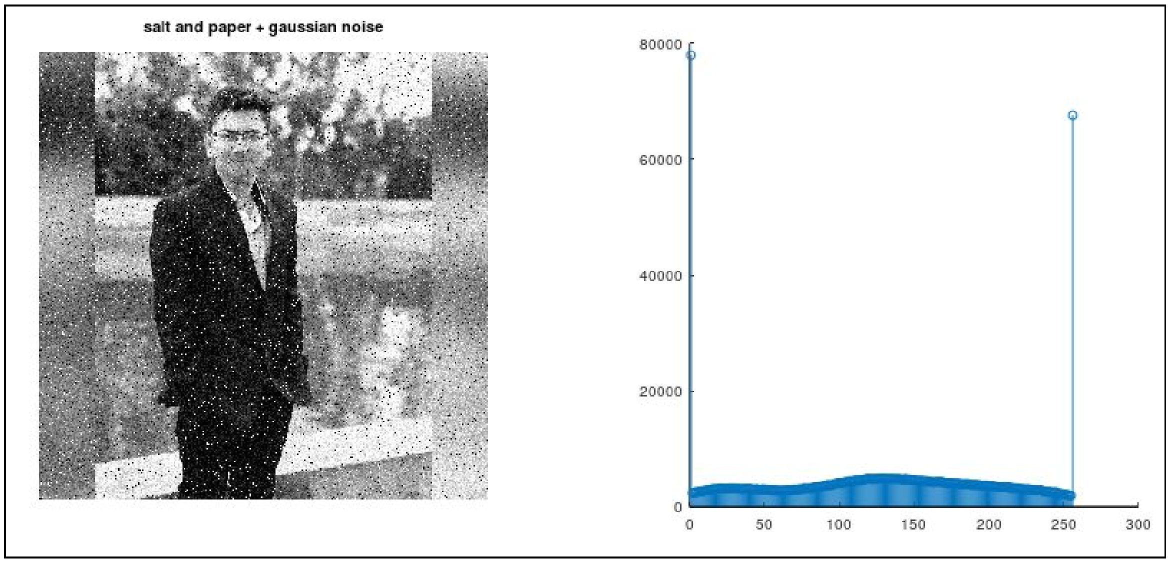


7. Gaussian plus salt and pepper noise.

Code :

```
gaussian_noisy=imnoise(r,'gaussian');
salt_pepper_and_gaussian=imnoise(gaussian_noisy,'salt & pepper');
subplot(1,2,1);
imshow(salt_pepper_and_gaussian);
title("salt and paper + gaussian noise");
x = imhist(salt_pepper_and_gaussian);
subplot(1,2,2);
stem(x);
imwrite(salt_pepper_and_gaussian,"salt_pepper_and_gaussian.jpg");
```

Output :

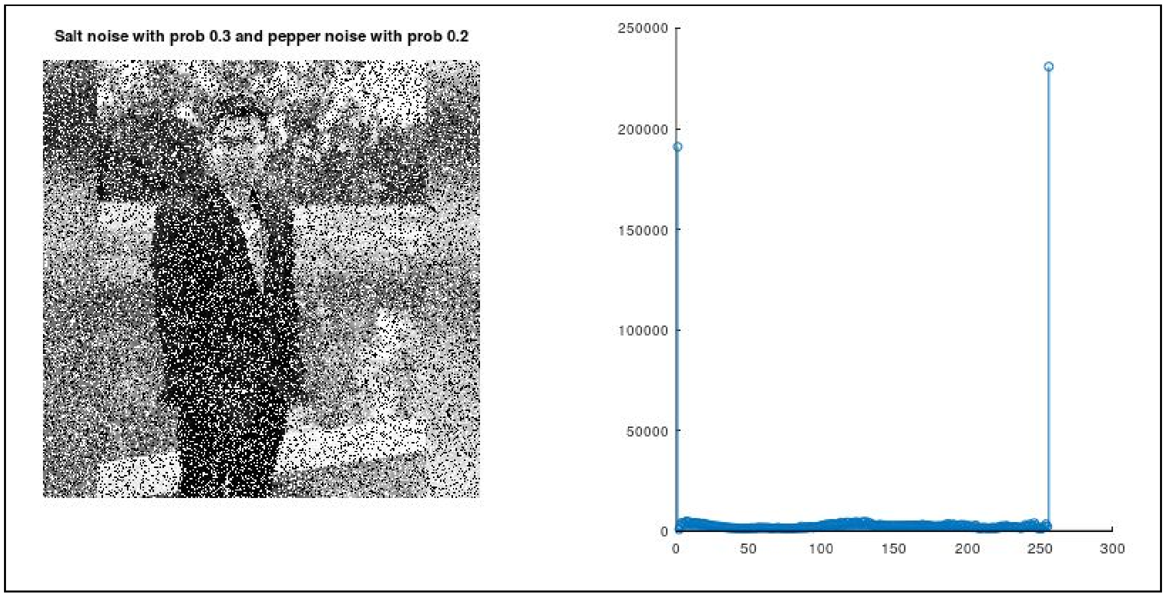


8. **Salt noise with probability 0.3 and pepper noise with probability 0.2.**

Code :

```
[salt_noise,y1]=salt_noise(r,0.3);
[salt_pepper,y2]=pepper_noise(salt_noise,0.2);
subplot(1,2,1);
imshow(salt_pepper);
title("Salt noise with prob 0.3 and pepper noise with prob 0.2");
subplot(1,2,2);
stem(y2);
imwrite(salt_pepper,"salt_0.3_pepper_0.2.jpg");
```

Output :

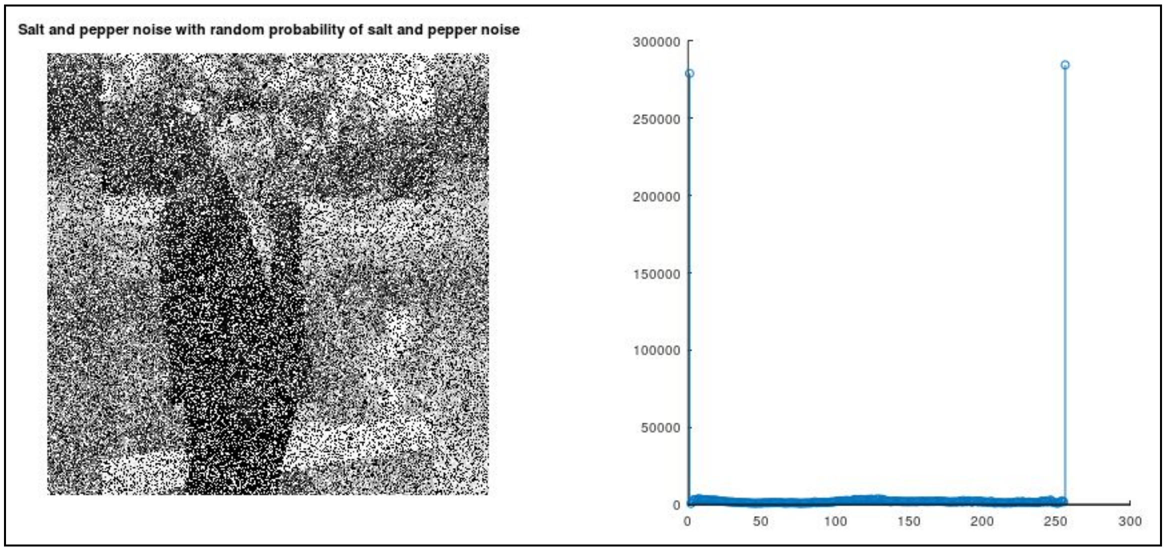


9. **Salt and pepper noise with random probability of salt and pepper noise.**

Code:

```
[noisy_im,y]=salt_and_pepper(r,rand());
subplot(1,2,1);
imshow(noisy_im);
title("Salt and pepper noise with random probability of salt and pepper noise")
subplot(1,2,2);
stem(y);
imwrite(noisy_im,"salt_pepper_random_prob.jpg");
```

Output:



Task 3 : Get information about imnoise and generate various noisy images.

- imnoise(A, type):-

  Add noise to the image.

- imnoise(A, “gaussian”,mean,variance):-

  Additive gaussian noise with mean and variance defaulting to 0 and 0.01.

- imnoise(A, “poisson”):-

  Creates poisson noise in the image using the intensity value of each pixel as mean.

- <u>imnoise(A, “salt & pepper”, density):-</u>

  Create "salt and pepper"/"lost pixels" in density*100 percent of the image. density defaults to 0.05.

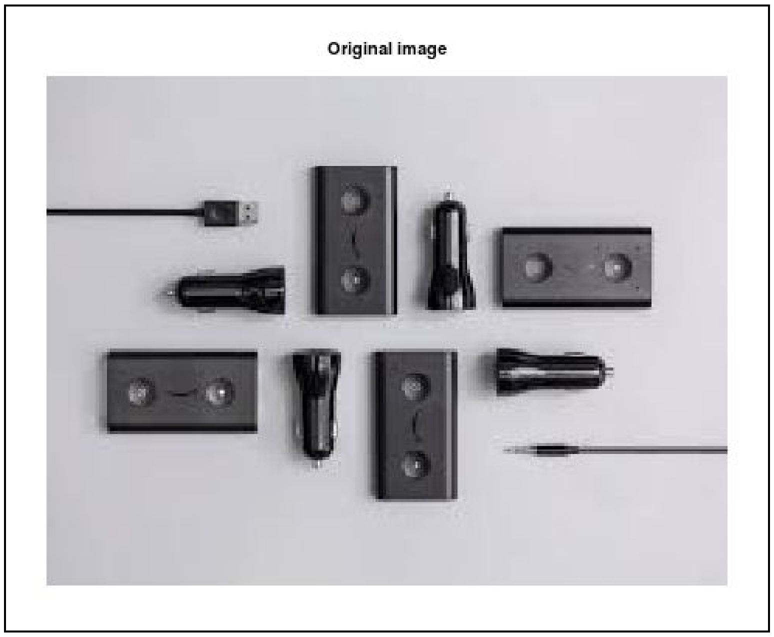
- <u>imnoise(A, “speckle”, variance):-</u>

  Multiplicative gaussian noise with B = A + A * noise with mean 0 and variance defaulting to 0.04.

Sample noise image using imnoise :

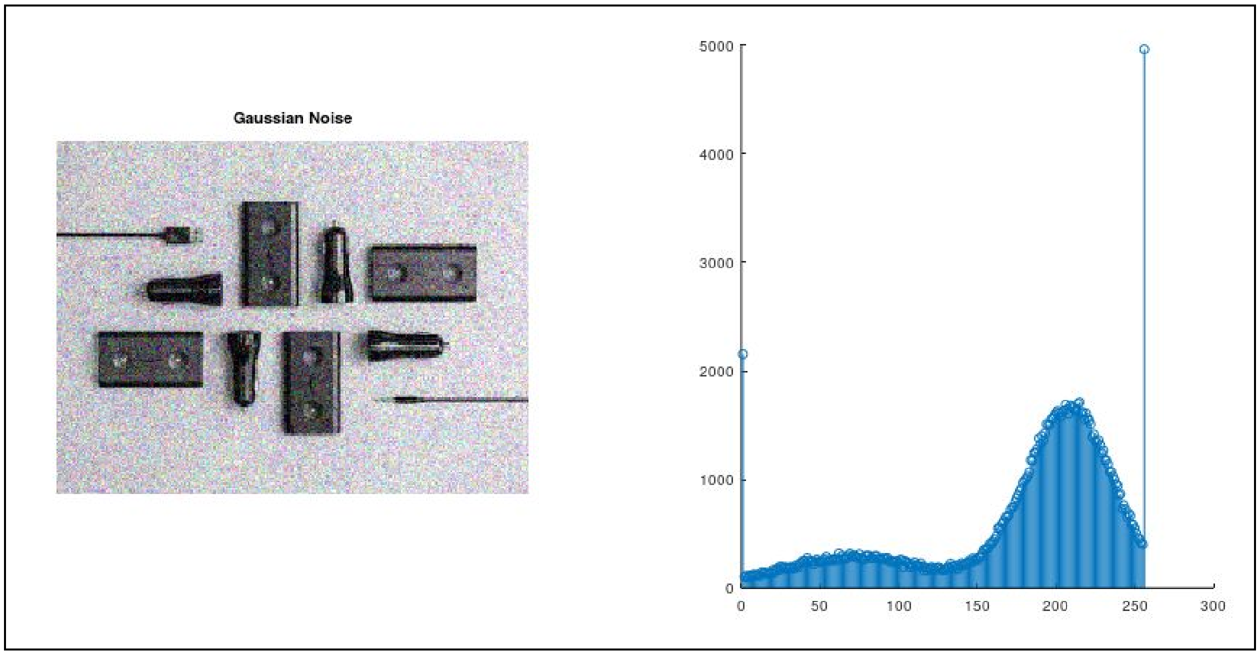
Original image :

```
r = imread("sample.jpg");
imshow(r);
title("Original image");
```



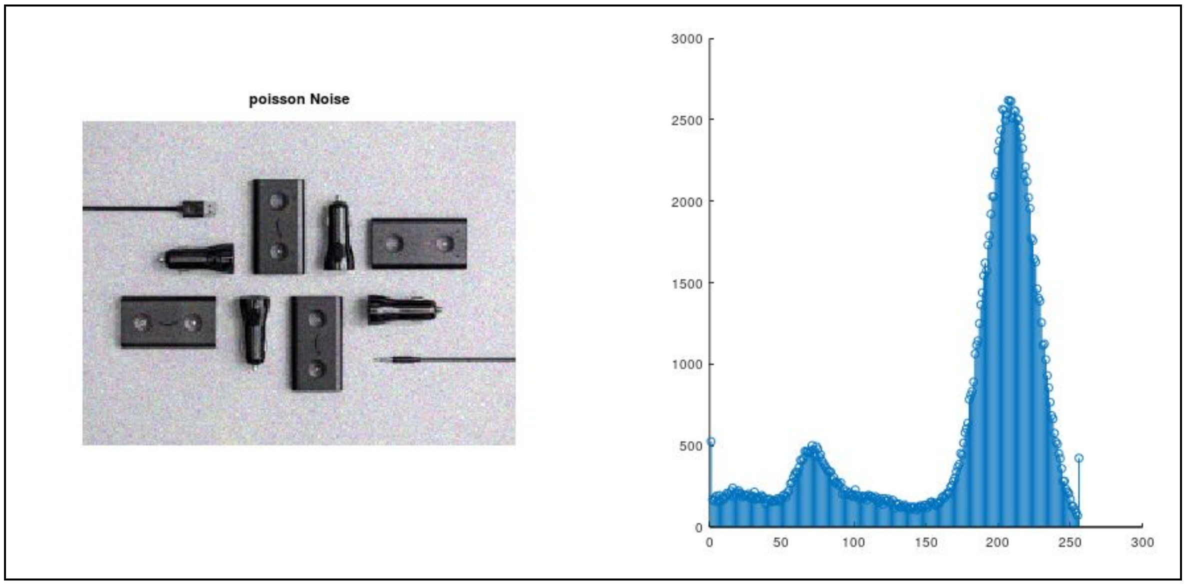
Gaussian noise:

```
gaussian_noise = imnoise(r,"gaussian",0,0.01);
subplot(1,2,1);
imshow(gaussian_noise);
title("Gaussian Noise");
x = imhist(gaussian_noise);
subplot(1,2,2);
stem(x);
```



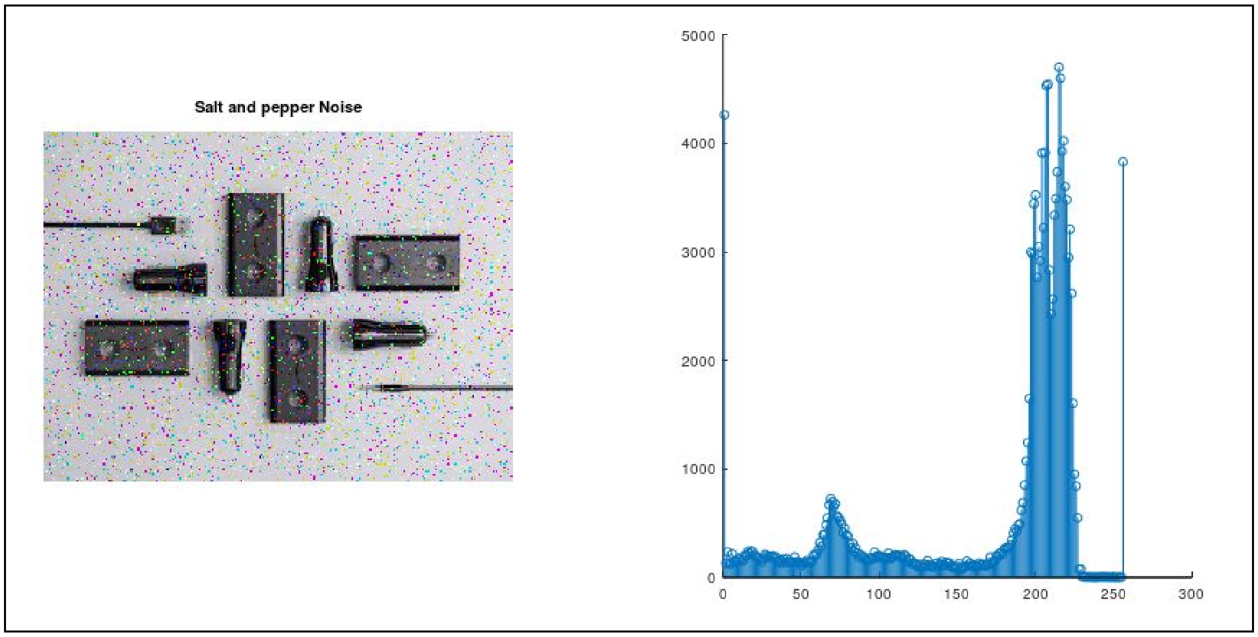
Poisson noise :

```
poisson_noise = imnoise(r,"poisson");
subplot(1,2,1);
imshow(poisson_noise);
title("poisson Noise");
x = imhist(poisson_noise);
subplot(1,2,2);
stem(x);
```



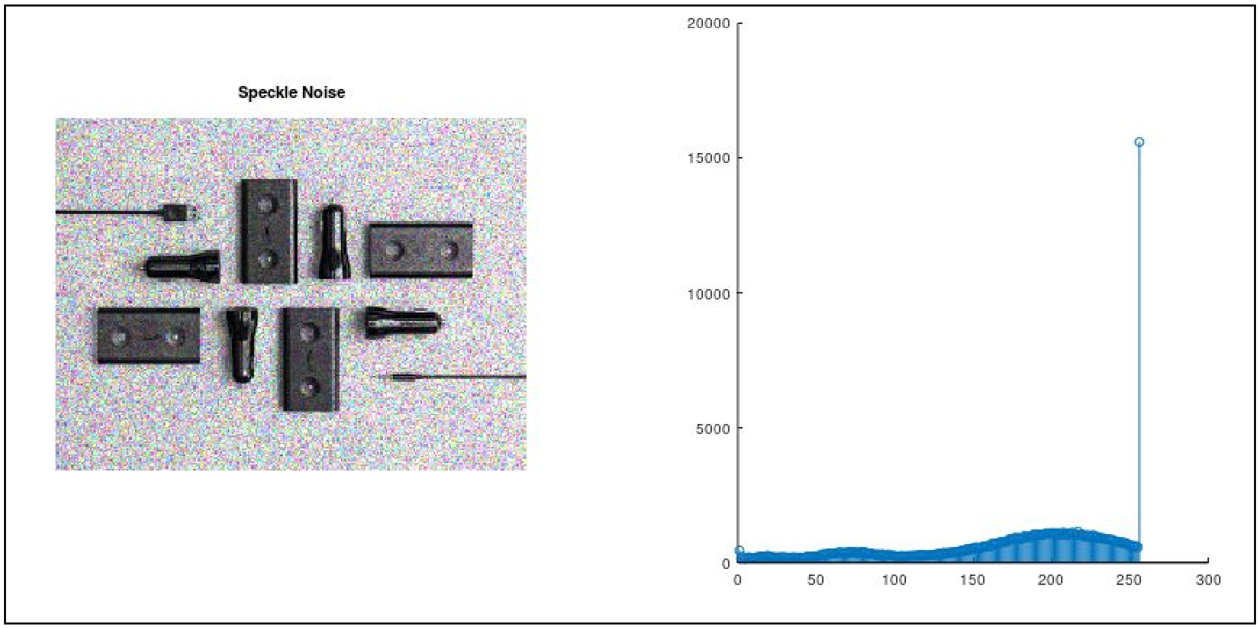
Salt and pepper noise :

```
salt_pepper_noise = imnoise(r,"salt & pepper");
subplot(1,2,1);
imshow(salt_pepper_noise);
title("Salt and pepper Noise");
x = imhist(salt_pepper_noise);
subplot(1,2,2);
stem(x);
```



Speckle noise :

```
speckle_noise = imnoise(r,"speckle");
subplot(1,2,1);
imshow(speckle_noise);
title("Speckle Noise");
x = imhist(speckle_noise);
subplot(1,2,2);
stem(x);
```


Speckle Noise