MATLAB CODE:-

Main Program:-

```
close all;
clear;
clc;


input_image=imread('linked.jpg');
figure;
imshow(input_image);
title('Input image');
im_gray=rgb2gray(input_image);
figure;
imshow(im_gray);
title('Input image after rgb 2 gray');
im_gray_1=im2double(im_gray);
figure;
imshow(im_gray_1);
title('Input gray image in double');
Noise_density = 0.50;
im_noised=imnoise(im_gray_1,'salt & pepper',Noise_density);
figure;
imshow(im_noised);
title(sprintf('Input noisy image with %d noise
density',Noise_density));
[p,q]=size(im_noised);
im_denoised=0.63*ones(p+16,q+16);
im_denoised(9:p+8,9:q+8)=im_noised;
epsilon=0.0001;
count0=0;
count1=0;
count2=0;
count3=0;
M=1;
N_init = 8;
im_denoised_pixels = zeros(p+16,q+16);
im_noised_pixels = zeros(p+16,q+16);
time_elapsed_per_itteration = zeros(1,10);
tic
e=1;
for z=1:
    im_denoised_pixels = zeros(p+16,q+16);
    im_noised_pixels = zeros(p+16,q+16);
    for j=9:q+8
        for i=9:p+8
            M = 1;
            N_init = 6;
            S_max = 2;
            N = N_init;
            while (im_denoised(i,j)==0)||(im_denoised(i,j)==1)
```

```
                [R_ij_M_matrix,index] =
R_ij_M(im_denoised,i,j,M);
                [T_min,T_max,T_min_max,PI,H,sigma,average_mu]
= membership_type_2(R_ij_M_matrix);

                lenght_R_ij_M_matrix = length(R_ij_M_matrix);
                ave_PI = sum(PI)/H;
                T_Threshold = T_max;
                o = 1;
                if ave_PI(H)>T_Threshold
                    count0=count0+1;
                    break
                elseif sigma==epsilon
                    im_denoised_pixels(i,j) = average_mu(1);
                    im_noised_pixels(i,j) = im_denoised(i,j);
                    count1=count1+1;
                    break
                end
                G=zeros(1,N);
                for x=1:lenght_R_ij_M_matrix
                    if  ave_PI(x)>=T_Threshold
                        count2=count2+1;
                        G(o)=R_ij_M_matrix(x);
                        o=o+1;
                    elseif
(R_ij_M_matrix(x)~=0)&&(R_ij_M_matrix(x)~=1)
                        count3=count3+1;
                        G(o)=R_ij_M_matrix(x);
                        o=o+1;
                    end
                    if o==N+1
                        break
                    end
                end
                neta=length(find(G));
                if (neta<N) && (M< S_max)
                    M=M+1;
                    continue
                elseif (neta<N) && (M== S_max)
                    N=N-1;
                    if N<1
                        S_max=S_max+1;
                        N=1;
                    end
                    continue
                end
                for k=1:(length(G)/2)
                    mean(k)=K_middle_mean_w(k,G);
                end
                mean_G=((sum(mean))/length(G));
                var_G=2.5*abs(G-mean_G);
```

```
                var_G=max(var_G);
                if var_G<=.01
                    var_G=.01;
                end
                w=gaussmf(G,[var_G,mean_G]);
                W=sum(w);
                weighted_G=w*G';
                im_denoised_pixels(i,j) = weighted_G/W;
                im_noised_pixels(i,j) = im_denoised(i,j);
                break
            end
        end
    end
    time_elapsed_per_itteration(z)=toc;
    im_denoised=(im_denoised-
im_noised_pixels)+im_denoised_pixels;
end
time_elapsed=toc;
im_denoised=im_denoised(9:p+8,9:q+8);
im_denoised_1 = im2uint8(im_denoised);
im_denoised_1=int16(im_denoised_1);
im_gray=int16(im_gray);
PSNR=10*log10((255*255)/((1/((p-10)*(q-
10)))*sum(sum((im_denoised_1(6:p-5,6:q-5)-im_gray(6:p-5,6:q-
5)).^2))));
fprintf('PSNR is %d\n',PSNR);
figure;
imshow(im_denoised);
title(sprintf('Denoised image with %d noise
density',Noise_density));
```

FUNCTION-1

```
function [vector,index, Window]=R_ij_M(image,i,j,M)
x=(i-M):(i+M);
y=(j-M):(j+M);
neighborhood_length = (2*M+1)^2;
Window=image(x,y);
vector=reshape(Window,[1,neighborhood_length]);
index=combvec(x,y);
end
```

FUNCTION-2

```
function
[T_min,T_max,T_min_max,PI,H,sigma,average_mu]=membership_type_
2(vector)
p=numel(vector);
H=(p+1)/2;
mu=zeros(1,H);
for q=1:H    mu(1,q)=K_middle_mean(q,vector);
```

```
end
average_mu=((sum(mu))/H)*ones(1,p);
lembda= 1.5*abs(vector-average_mu).^2;
mu_Mat=repmat(mu,p,1)';
lembda_vector=repmat(vector,H,1);
sigma=K_middle_mean(H,lembda);
if sigma < 0.0001
    sigma=0.0001;
end
PI= exp(-0.5*((lembda_vector-mu_Mat)/sigma).^2);
T_max=max(max(PI));
T_min_max=min(max(PI));
T_min=max(min(PI));
end
```

FUNCTION-3

```
function mean=K_middle_mean_w(k,Vector)
Lenght_of_Vector = numel(Vector);
Vector=sort(Vector);
if mod(Lenght_of_Vector,2) == 1
    half_lengh = 0.5*(Lenght_of_Vector+1);
    factor = 1/(2*k - 1);
    K_middle_vector=Vector((half_lengh-k+1):(half_lengh+k-1));
    sum_of_element=sum(K_middle_vector);
else
    half_lengh=0.5*Lenght_of_Vector;
    factor=1/(2*k);
    K_middle_vector=Vector((half_lengh-k+1):(half_lengh+k));
    sum_of_element=sum(K_middle_vector);
end
mean=factor*sum_of_element;
end
```

FUNCTION-4

```
function mean=K_middle_mean(k,Vector)
Lenght_of_Vector = numel(Vector);
Vector=sort(Vector);
if mod(Lenght_of_Vector,2) == 1
    half_lengh = 0.5*(Lenght_of_Vector+1);
    factor = 1/(2*k - 1);
    K_middle_vector=Vector((half_lengh-k+1):(half_lengh+k-1));
    sum_of_element=sum(K_middle_vector);
end
mean=factor*sum_of_element;
end
```