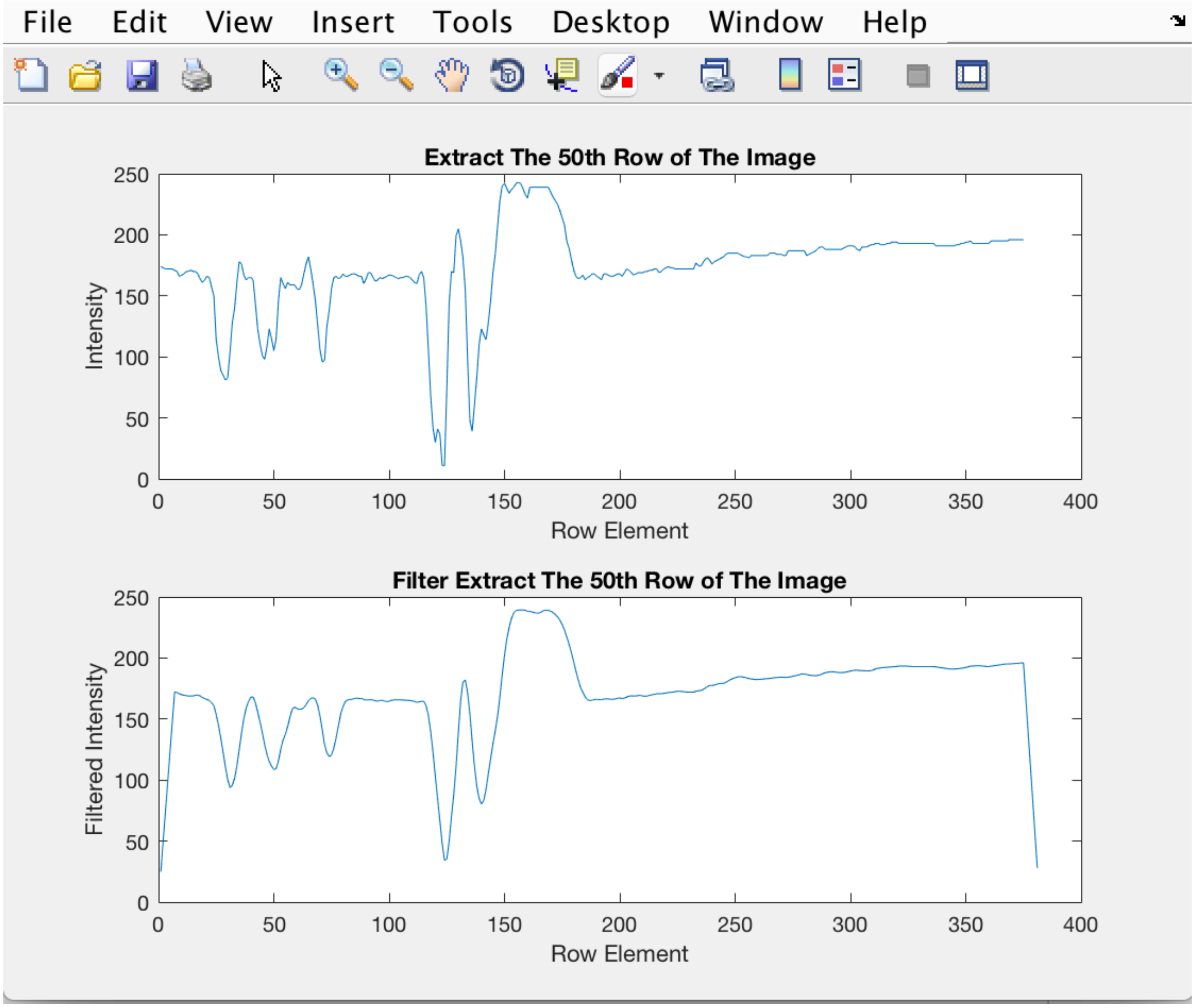
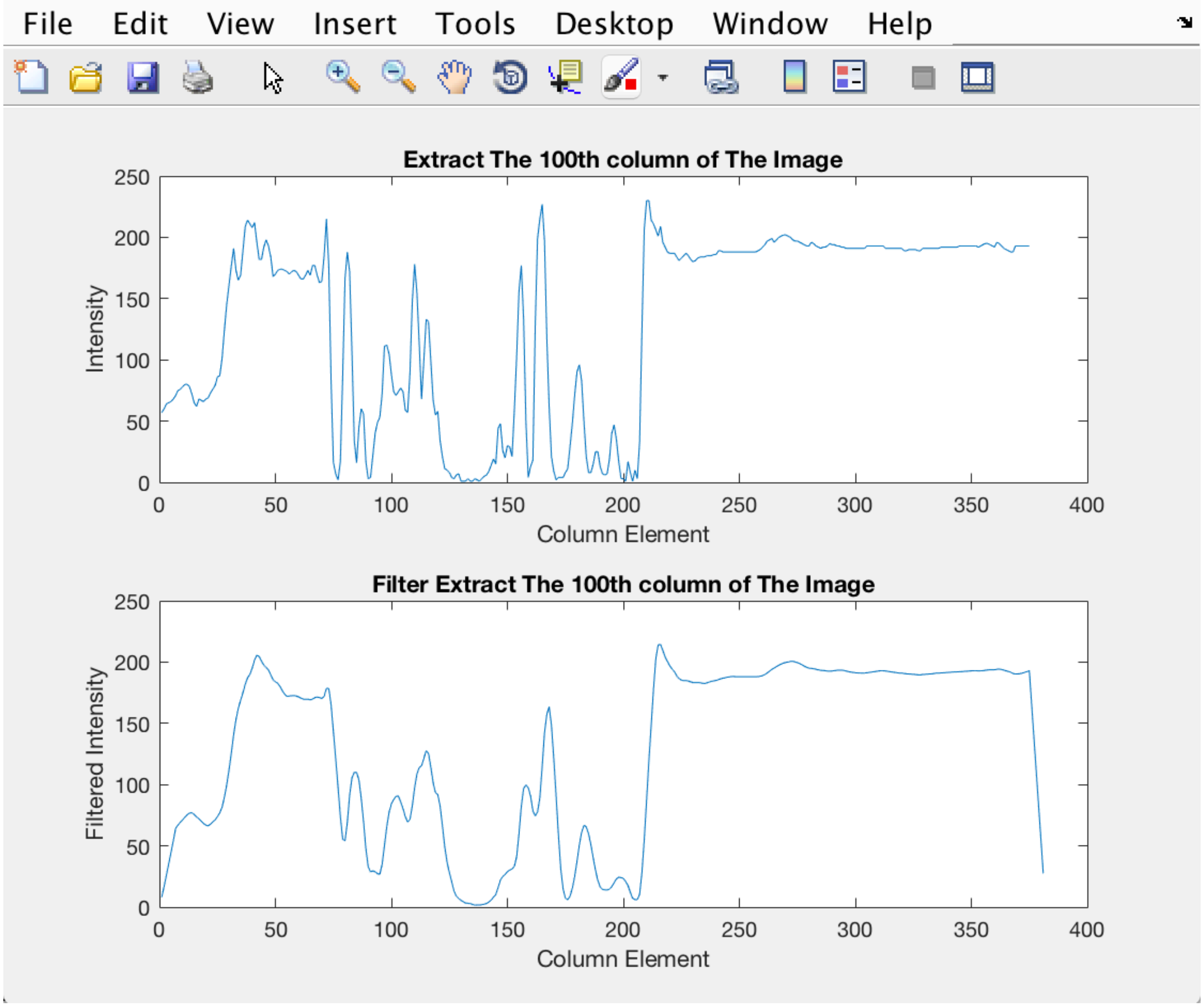
2.



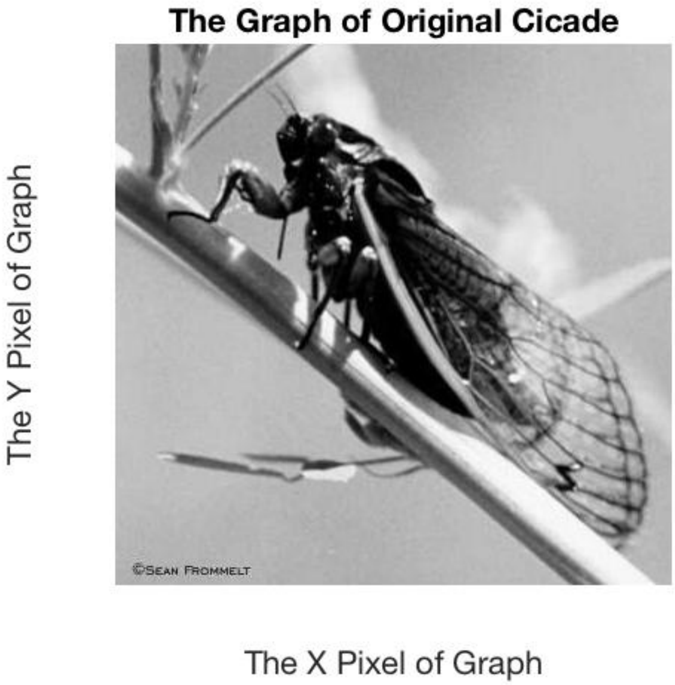
For 100th column:

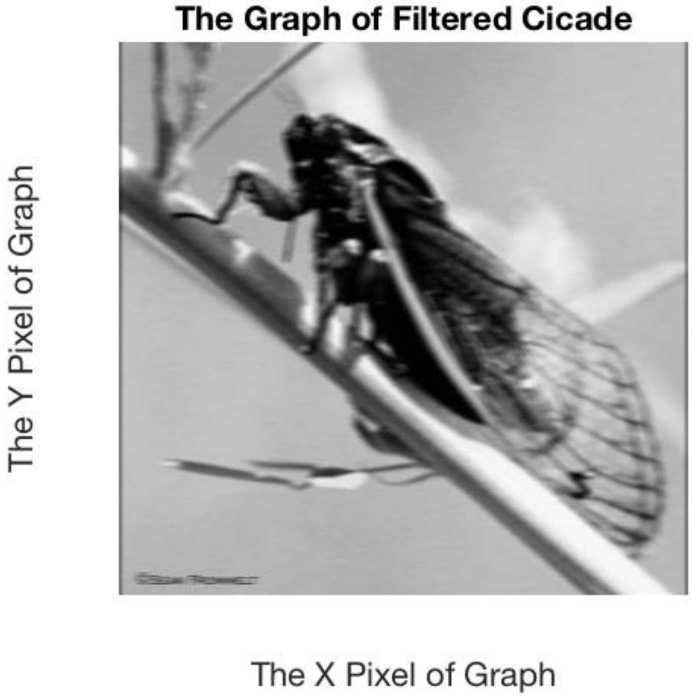
```
clear all;
close all;
clc;

load('ip2_images');

column = cicada(100,:); %Extracts 100th row
ave_filt = ones(1,7)/7; %Calculate filter coefficients
filt_column = conv(column,ave_filt); %Filter 100th column

subplot(2,1,1); % plot the unfiltered signals
plot(column),
title('Extract The 100th column of The Image'),
xlabel('Column Element'),
ylabel('Intensity'),

subplot(2,1,2); %plot the filtered signals
plot(filt_column),
title('Filter Extract The 100th column of The Image')
xlabel('Column Element'),
ylabel('Filtered Intensity'),
```



The filtered output is smoother than the input.

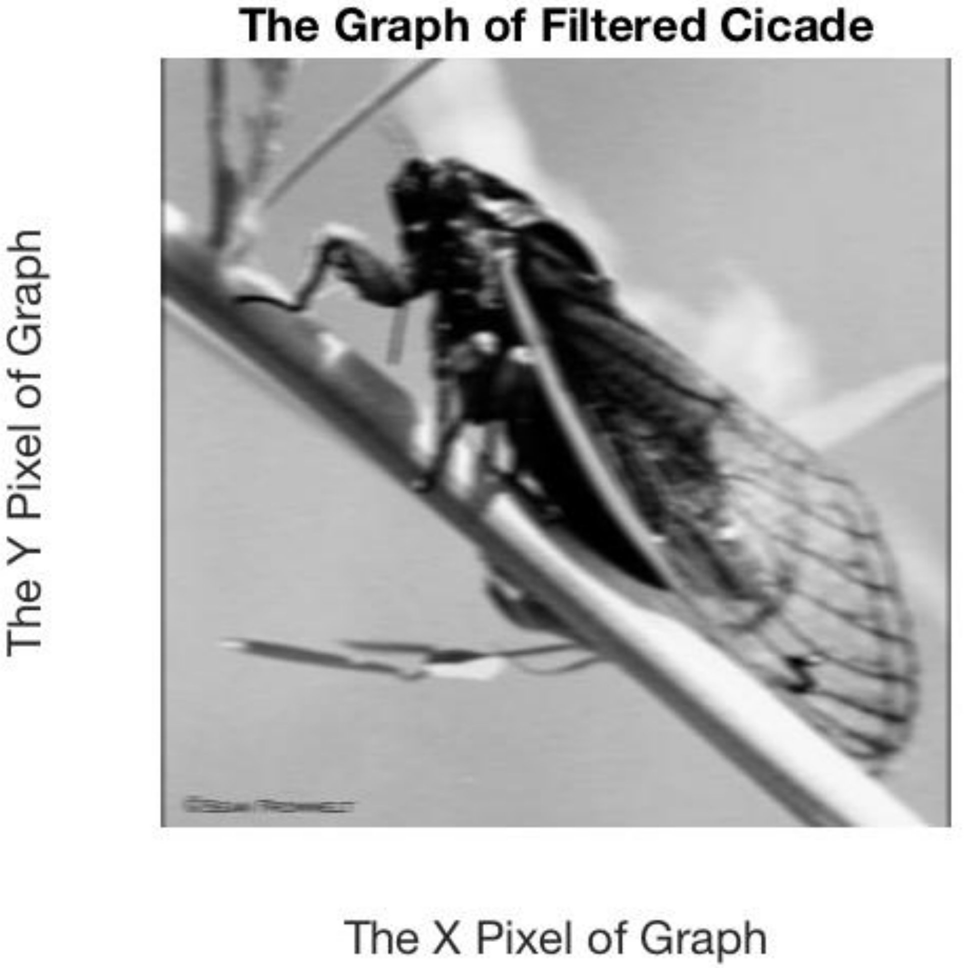
2.1



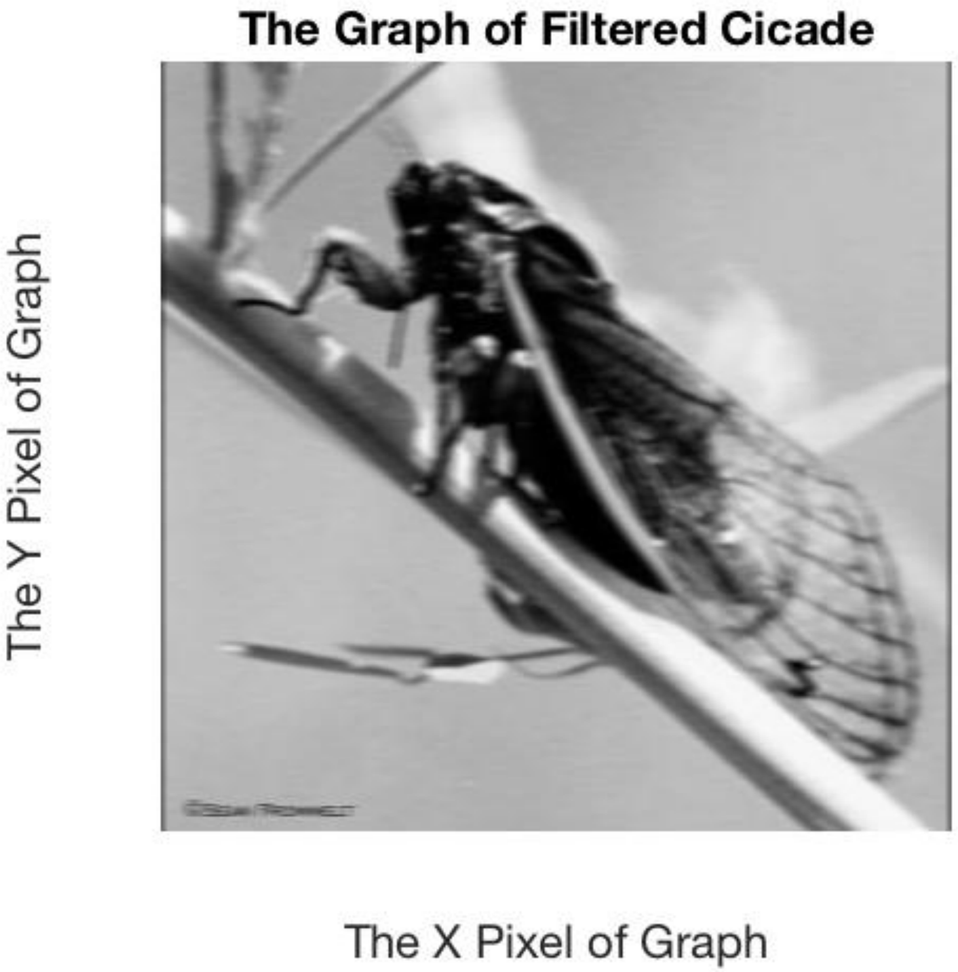


The filtered image looks like a blurred cicade's image. The quality of the image is not very clear, and is blurred throughout the image.

Filtering row only:



Filtering row and column:

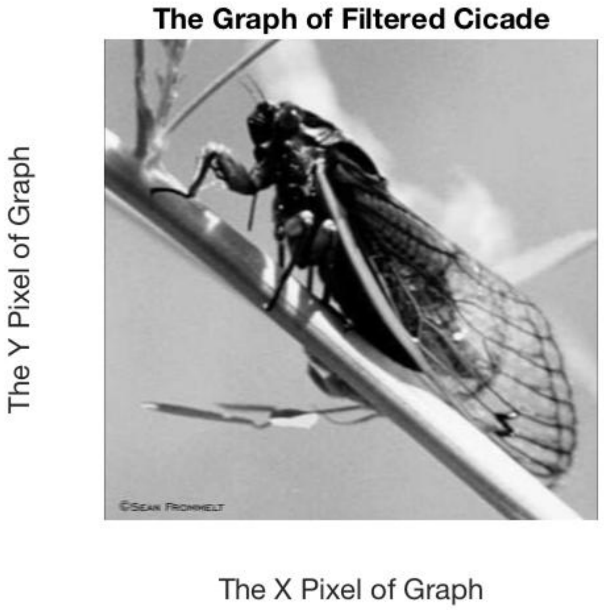


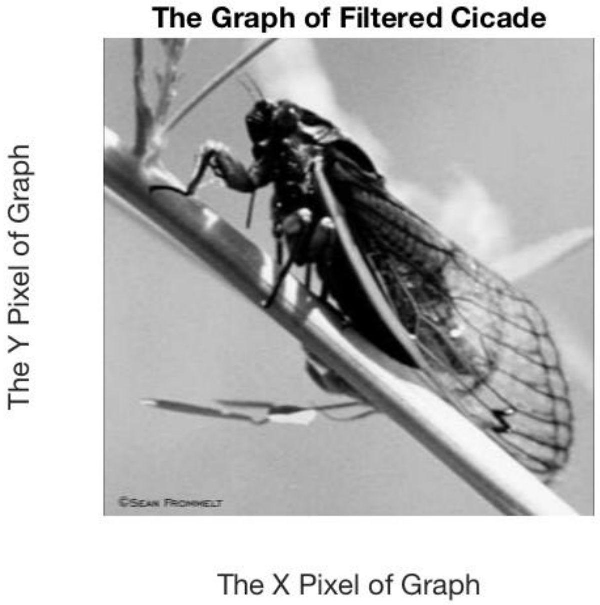
Both the filtered images look's the same blurry.

For column filtering following modification in Matlab code was used:

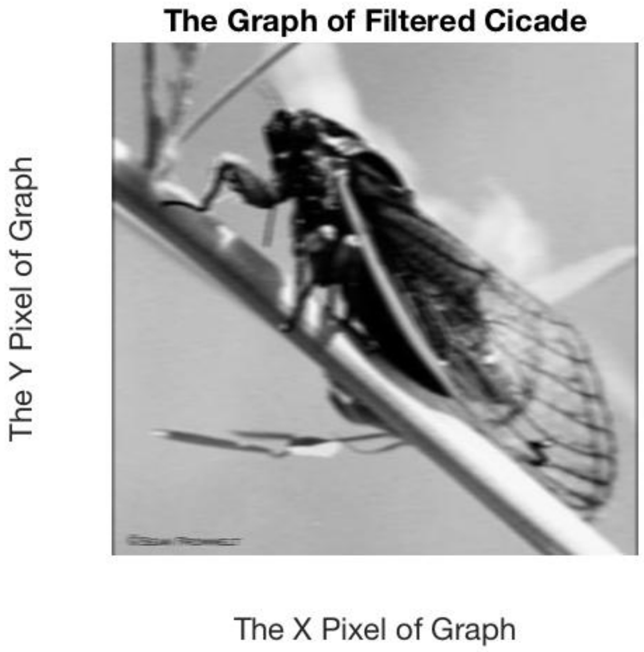
```
function y_column = filtcolumn(image,filt)

[row,column] = size(image);

for y = 1 : column
    y_column(:,y) = conv(image(:,y),filt);
    y_column(:,y) = round(y_column(:,y));
end

length_filt = length(filt); %find filter half-length
half_len = floor(length_filt/2);
y_column = y_column(row+half_len:half_len+1,:); %extract the c pixels in the center of the columns
```

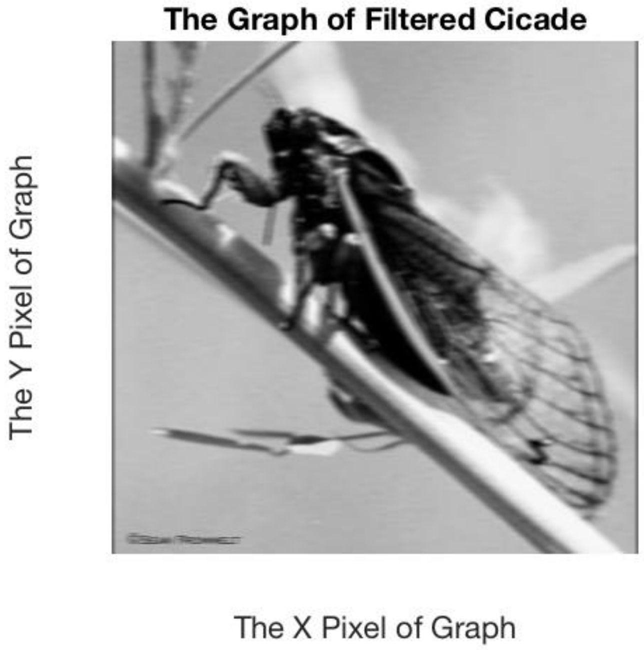
For 3 point averaging filter:



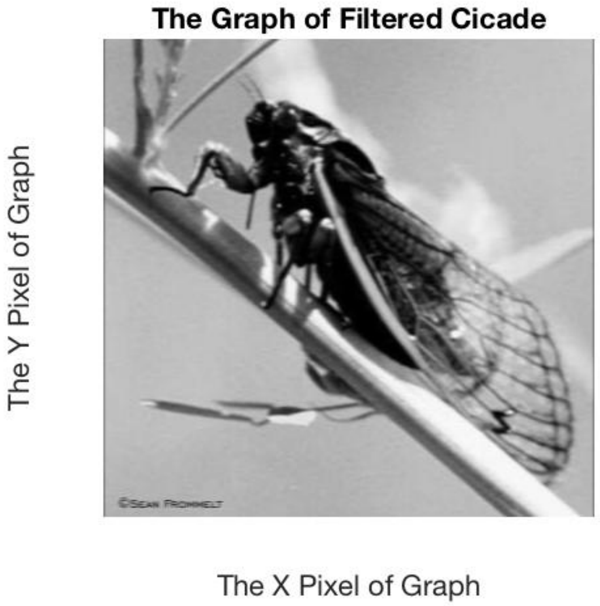


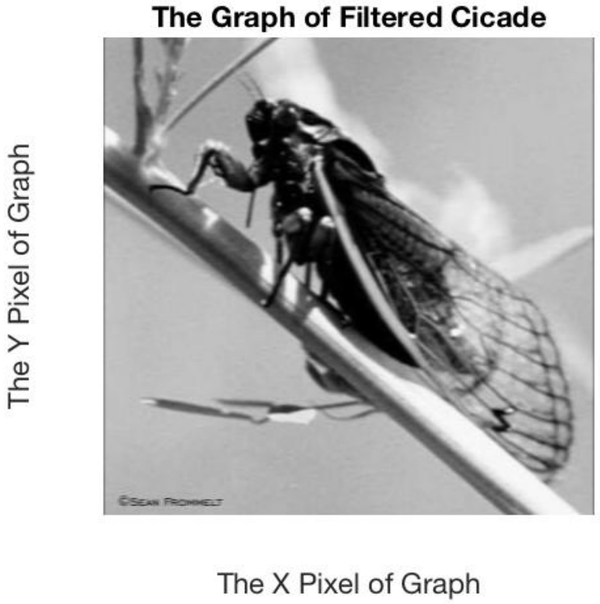
For 7 point averaging filter:


The Graph of Filtered Cicade
The Y Pixel of Graph
The X Pixel of Graph


The Graph of Filtered Cicade
The Y Pixel of Graph
The X Pixel of Graph

For [1 2 4 2 1]/10 :



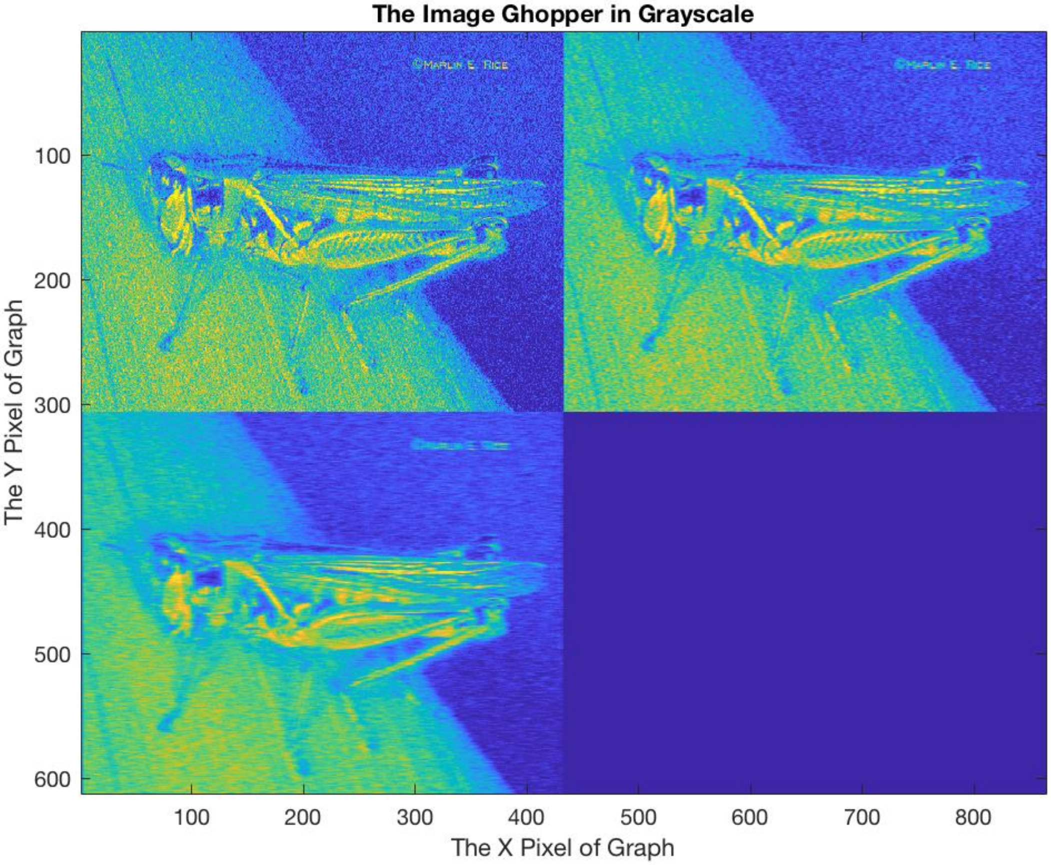


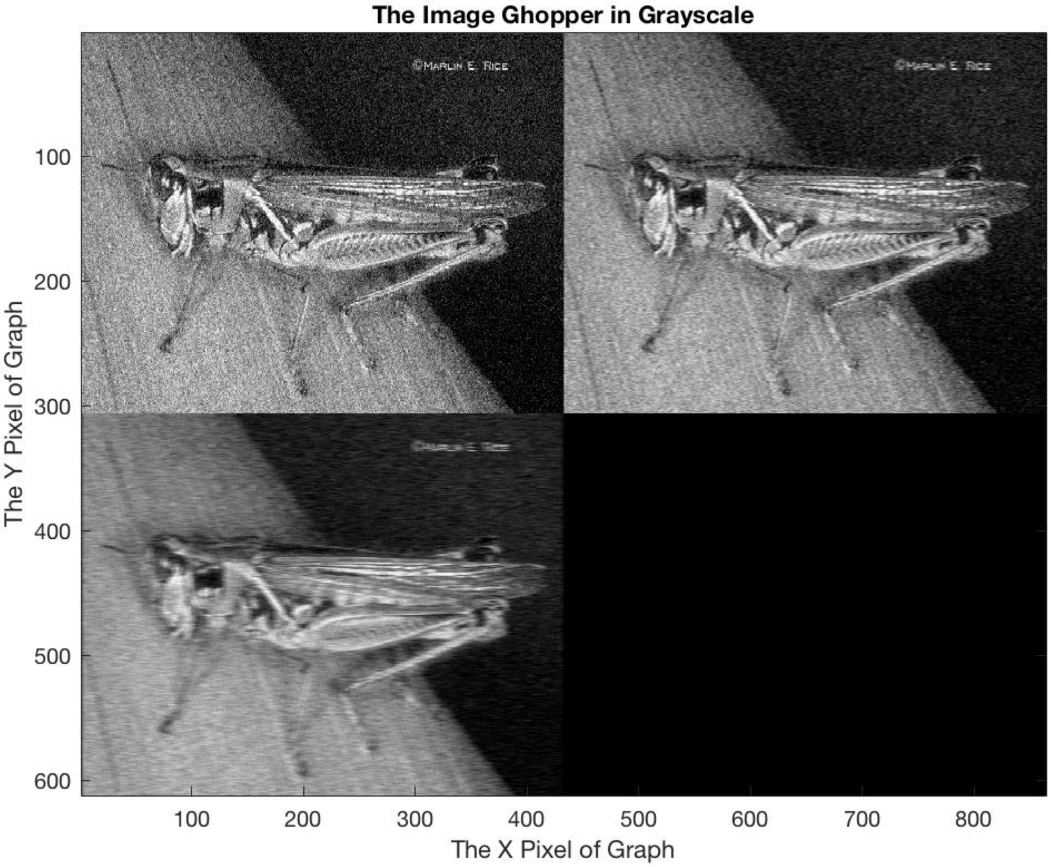
Since from the above images all the filtered images are blurred. Hence, all the above are low-pass filter.

A low pass filter is a filter which blurs the pixels of the image, while a high-pass filter intensifies it.

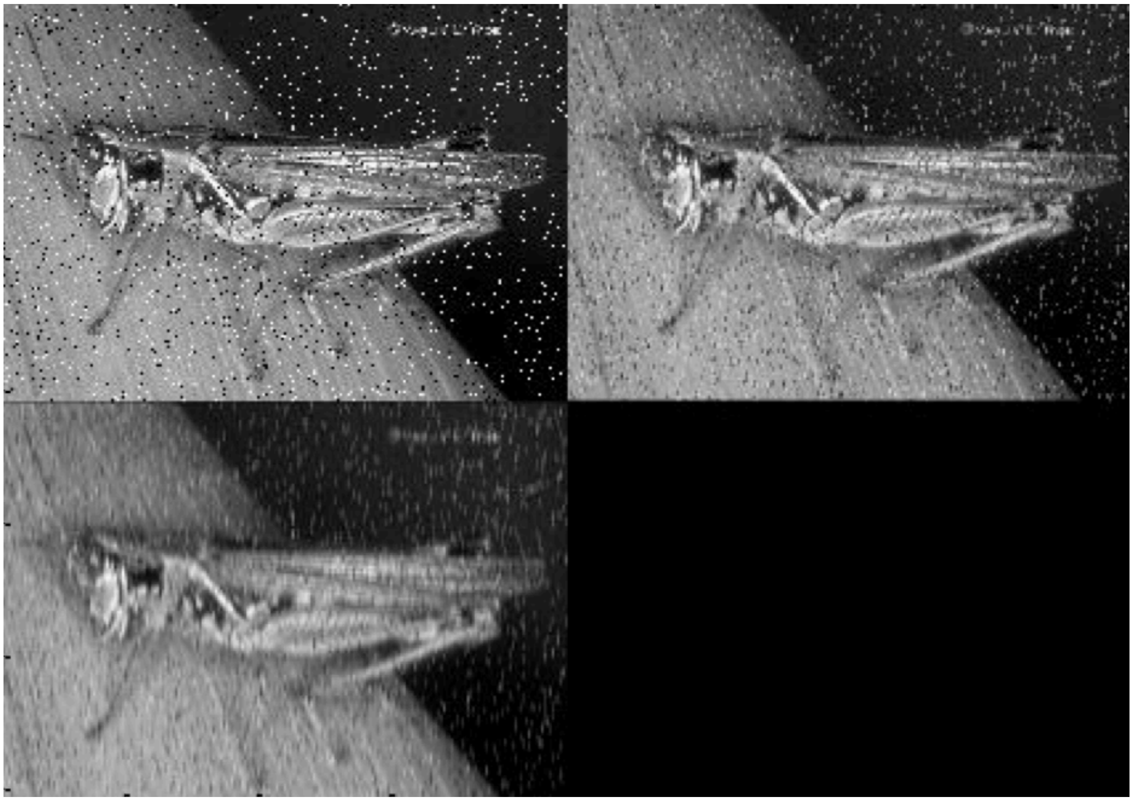
2.2

©Marlin E. Rice


The Image Ghopper in Grayscale
The Y Pixel of Graph
100
200
300
400
500
600
The X Pixel of Graph
100
200
300
400
500
600
700
800
©Marlin E. Rice


The Image Ghopper in Grayscale
The Y Pixel of Graph
100
200
300
400
500
600
100
200
300
400
500
600
700
800
The X Pixel of Graph
©MARLIN E. RICE
©MARLIN E. RICE
©MARLIN E. RICE

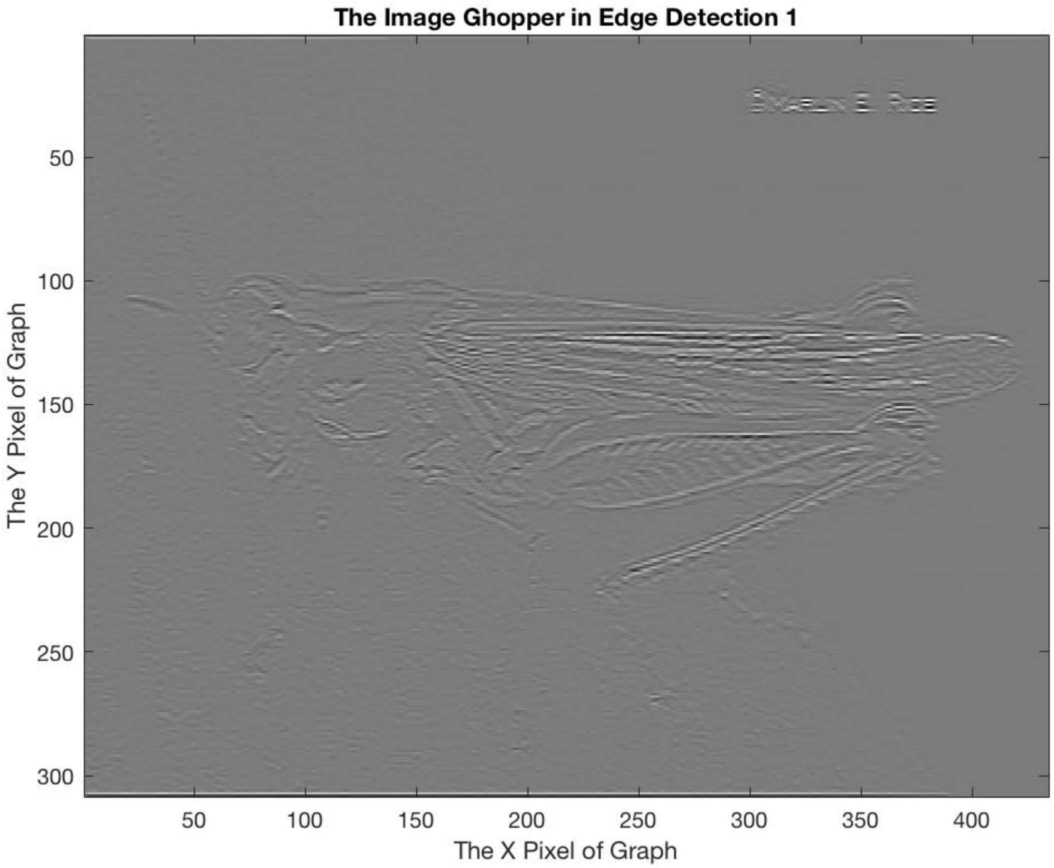
©MARLIN E. RICE



Well from the above images, I feel that filtering worked same for all the type of noises.

2.3

For image using filter (a):

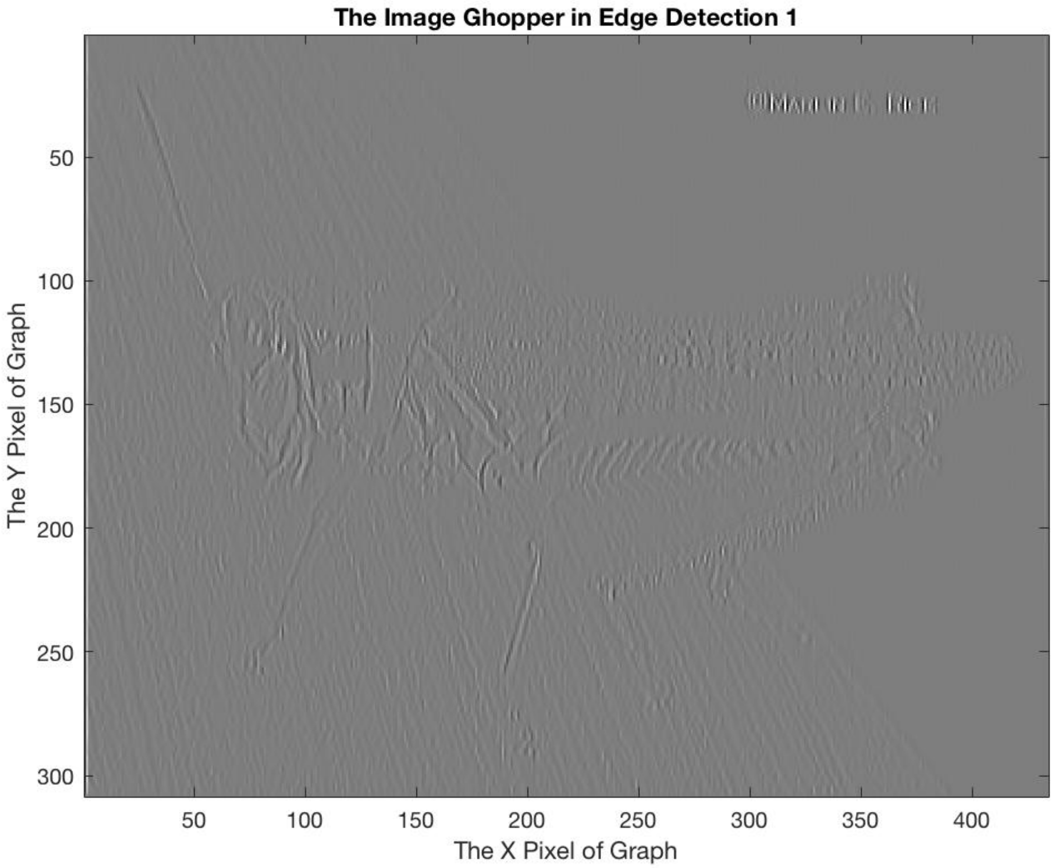


The horizontal edges were highlighted in this case.

For image using filter (c):

The following Matlab code was used:

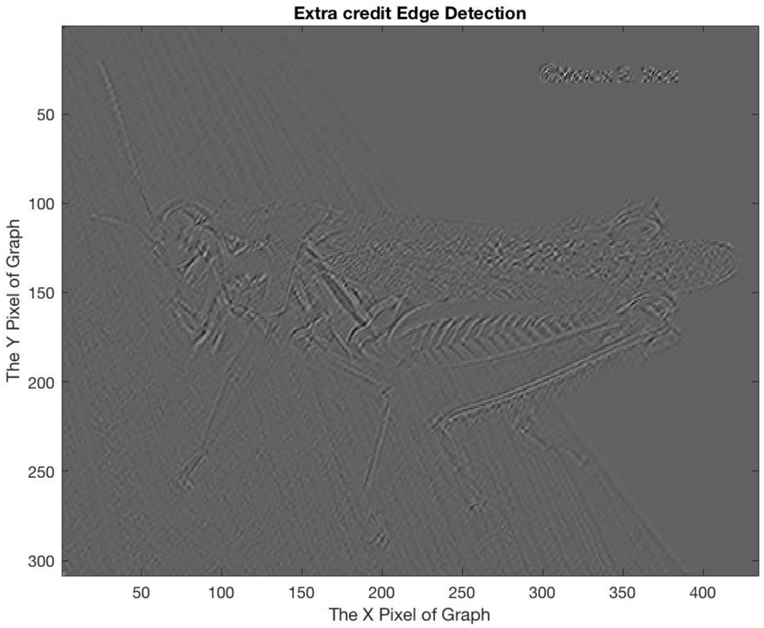
```
clear all;
close all;
clc;

load('ip2_images');

wc = [-1,2,-1;-1,2,-1;-1,2,-1]; % filter (c)

y_c=conv2(ghopper,wc);
y_c=round(y_c);
imagesc(y_c);
colormap('gray(256)');
title('Extra credit Edge Detection'),
xlabel('The X Pixel of Graph'),
ylabel('The Y Pixel of Graph'),
```

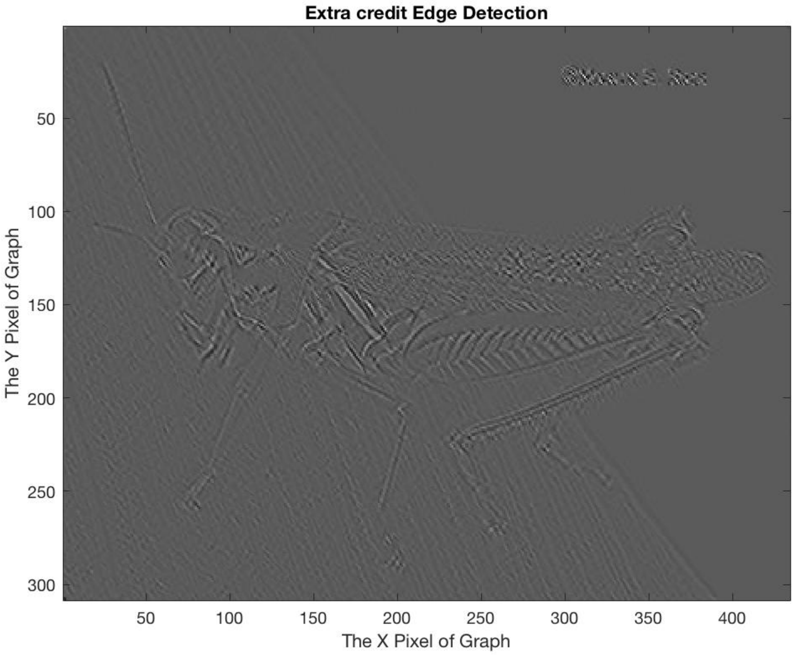


The vertical edges were highlighted in this case.

Comparing the above images, (a) has horizontal edges, while (c) has vertical edges.

Extra Credit:


Extra credit Edge Detection
The Y Pixel of Graph
50
100
150
200
250
300
The X Pixel of Graph
50
100
150
200
250
300
350
400


Extra credit Edge Detection
The Y Pixel of Graph
50
100
150
200
250
300
The X Pixel of Graph
50
100
150
200
250
300
350
400