Luis Hernandez Aguirre	CM
Fox Warner	CM

Lab 5: Image Filtering – Worksheet

ECE180: Introduction to Signal Processing

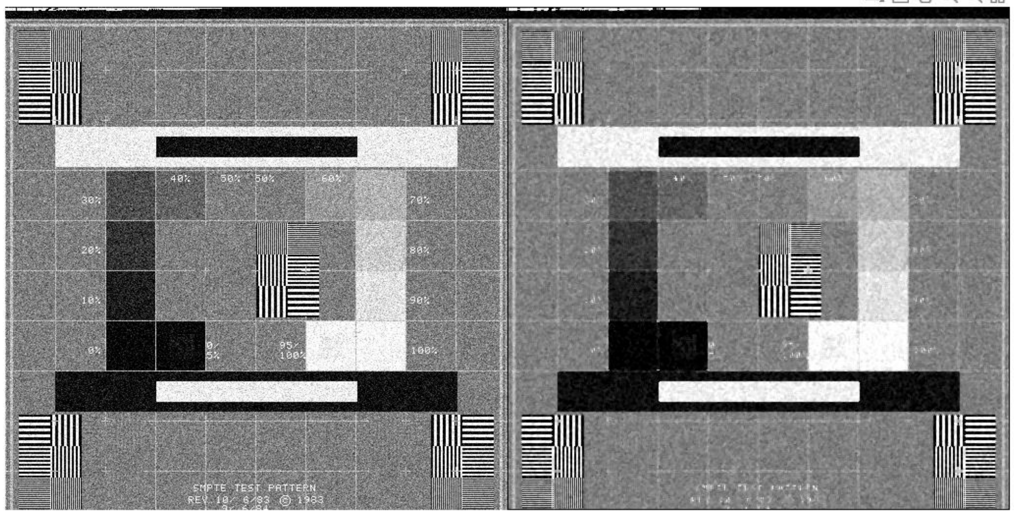
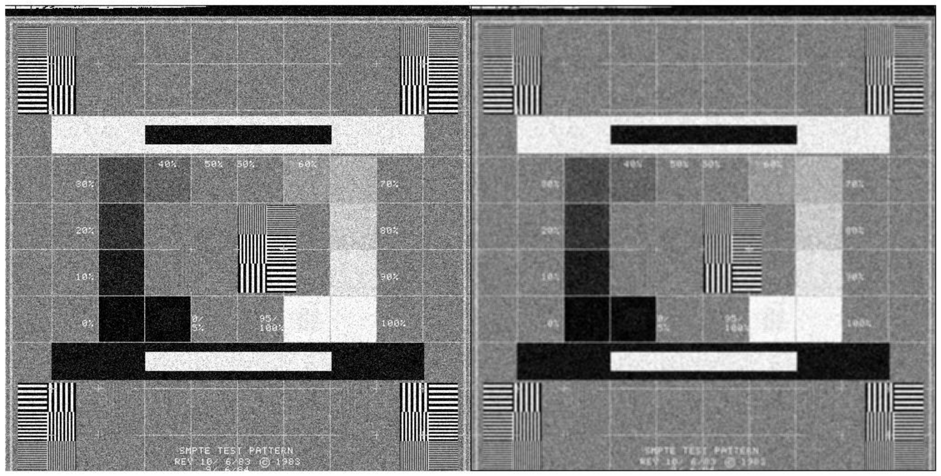
1.3 🗎

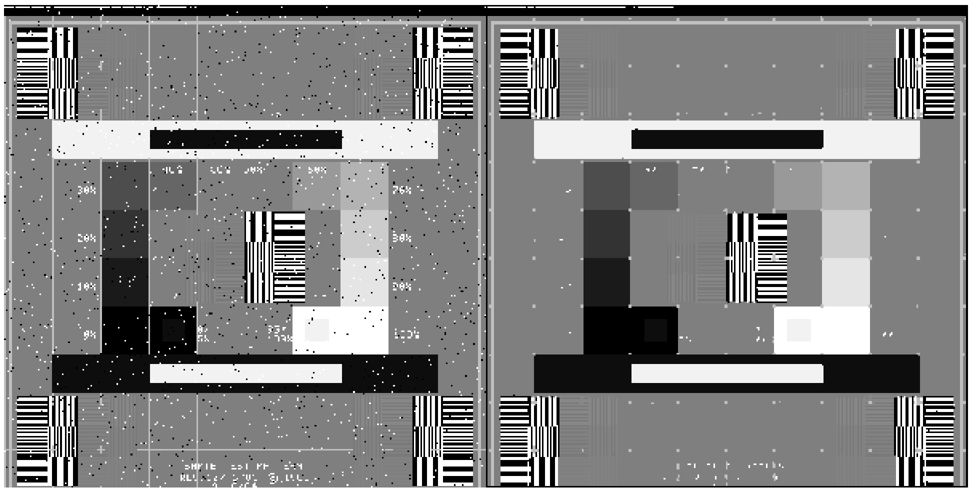
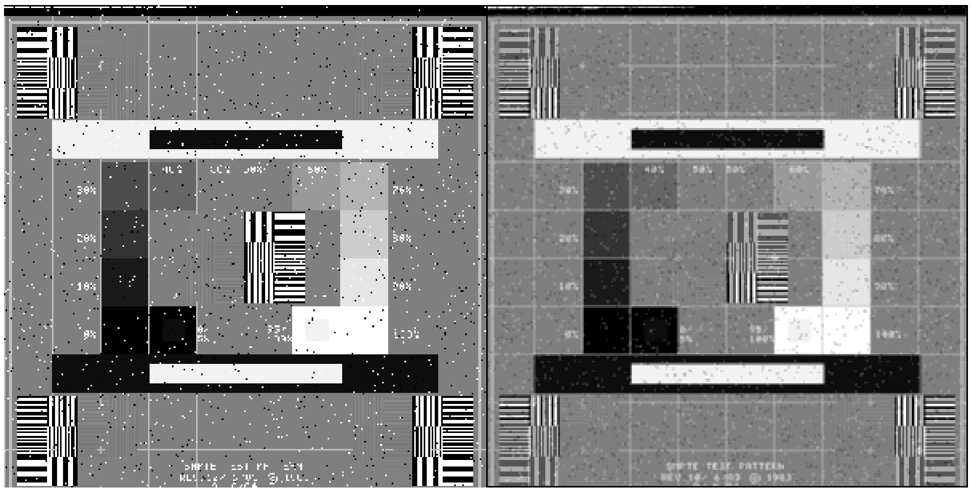
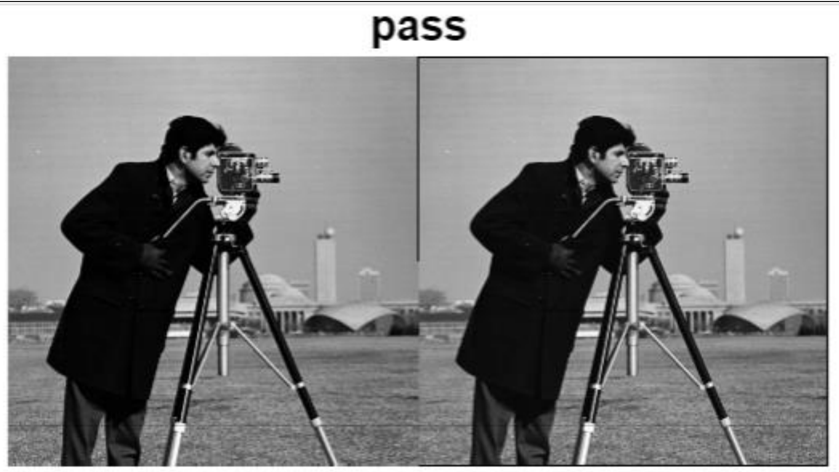
```
function [y] = conv3x3(x,h)


x = double(x);

y = x;

y(:) = 0;

y = uint8(y)

sz = size(x);


for i = 2:sz(1)-1

    for j = 2:sz(2)-1

        rim = ...

        [x(i-1, j-1) x(i, j-1) x(i+1, j-1);

         x(i-1, j)   x(i,j)    x(i+1, j);

         x(i-1, j+1) x(i,j+1)  x(i+1, j+1);]


        rim = rim.*h;

        rim = rim(:);

        s = sum(rim, "all")

        y(i,j) = uint8(s)

    end

end


imshow(y)
```

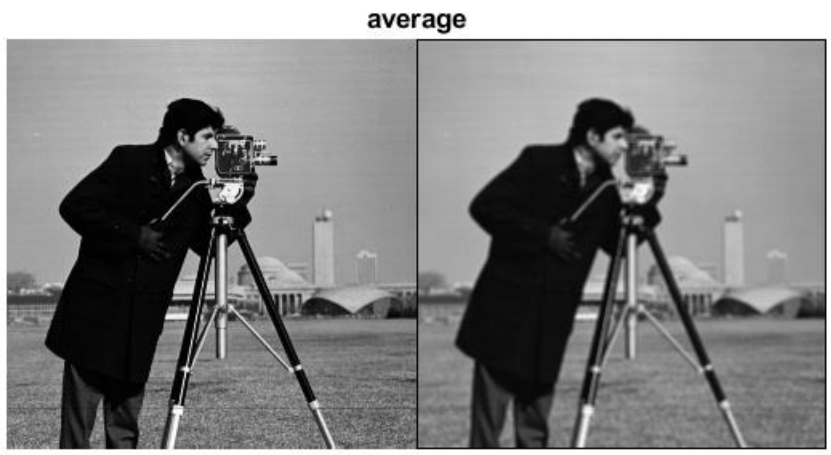
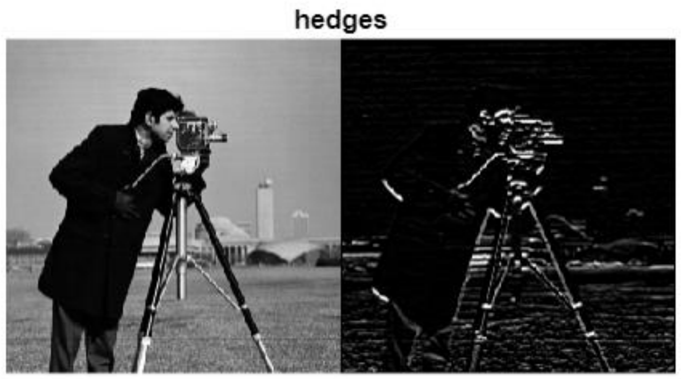
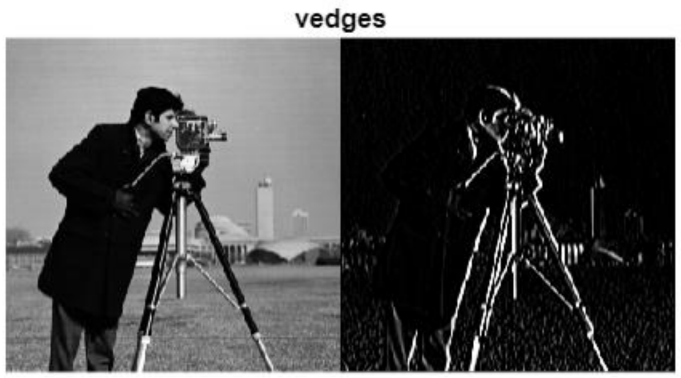
2.3 🗎

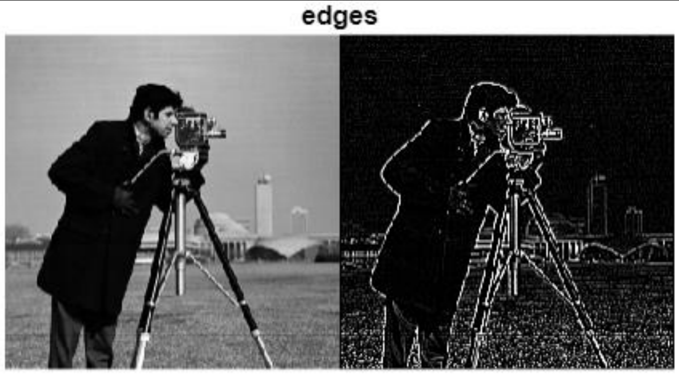
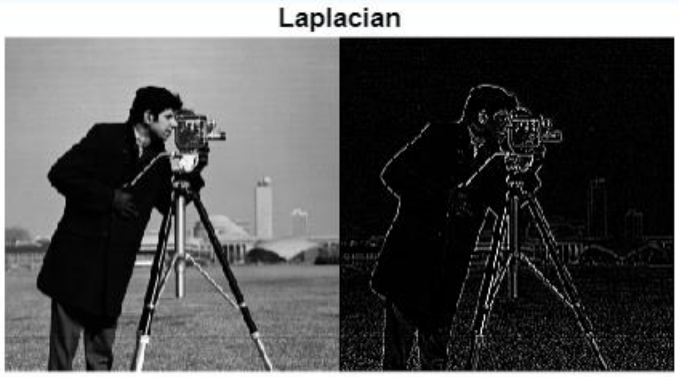
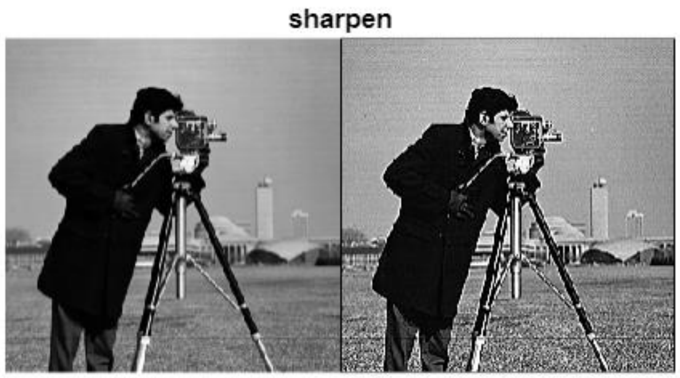
```
function [y] = med3x3(x)
```

	```
	x = double(x);

	y = x;

	y(:) = 0;

	y = uint8(y);

	sz = size(x);


	for i = 2:sz(1)-1

	    for j = 2:sz(2)-1

	        rim = ...

	        [x(i-1, j-1) x(i, j-1) x(i+1, j-1);

	         x(i-1, j)   x(i,j)    x(i+1, j);

	         x(i-1, j+1) x(i,j+1)  x(i+1, j+1);];


	        rim = rim(:);

	        rim = sort(rim,'descend');

	        s = rim(5);

	        y(i,j) = uint8(s);

	    end

	end


	imshow(y)

	end
	```

3.4	 40% 50% 50% 60% 30% 70% 20% 80% 10% 90% 0% 0/5% 95/100% 100% SMPTE TEST PATTERN REV 10/ 6/83 © 1983
3.7	 40% 50% 50% 60% 30% 70% 20% 80% 10% 90% 0% 0/5% 95/100% 100% SMPTE TEST PATTERN REV 10/ 6/83 © 1983
3.8	The median filter lost less color than the convolution filter. The striped boxes in the corners lost less of their distinction as the convolution filter blurred and meshed the black and white to make greyer tones. Similarly, across the entire picture the convolution filter is much blurrier. But the median filter renders any letters illegible.

3.9(a) 🖼	
3.9(b) 🖼	
3.10 ✎	The media filter removed all the pepper, leaving salt only in the corners of columns and rows. Meanwhile the convolution filter just blurred everything. The salt and pepper blend in a bit more with the background, but they all remain visible.
4.2(a) 🖼 ✎	 Pass did nothing to the picture, it remains unchanged.

4.2(b)	 Average blurs the image.
4.2(c)	 Hedges turns the image basically black, it outlines everything by reflecting any white items onto the black items to the right.
4.2(d)	 Does the same thing as hedges, except the reflection is downwards

4.2(e)	 Edges turns white spaces of white into black, and turns any black pixels neighboring a white pixel into white.
4.2(f)	 It does the same thing as edges, except the black pixels are less effected by the surrounding white pixels.
4.2(g)	 This appears to sharpen the image. It just turns greyish spots into white or black.

4.3	`function [y] = adjust(kernel, imageTitle, fontSize)`
	
	`x = imread(imageTitle);`

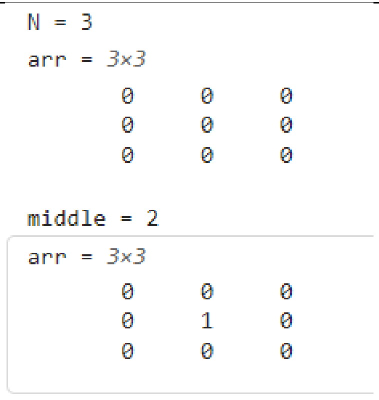
```
switch(kernel)

    case 'average'

        h = ...

            [1 1 1;

            1 1 1;

            1 1 1] * (1/9);

    case 'hedges'

        h = ...

            [-1 0 1;

            -1 0 1;

            -1 0 1;];

    case 'vedges'

        h = ...

            [ -1 -1 -1;

            0 0 0;

            1 1 1];

    case 'edges'

        h = ...

            [-1 -1 -1;

            -1 8 -1;

            -1 -1 -1];

    case 'Laplacian'

        h = ...

            [0 -1 0;

            -1 4 -1;

            0 -1 0];

    case 'sharpen'
```

```
        h = ...

            [0 -1 0;

            -1 5 -1;

            0 -1 0];

    case 'pass'

        h = ...

            [0 0 0;

            0 1 0;

            0 0 0];

end

y = conv3x3(x,h);



figure;

imshowpair(x,y, 'montage');

title(kernel, FontSize = fontSize)

end
```

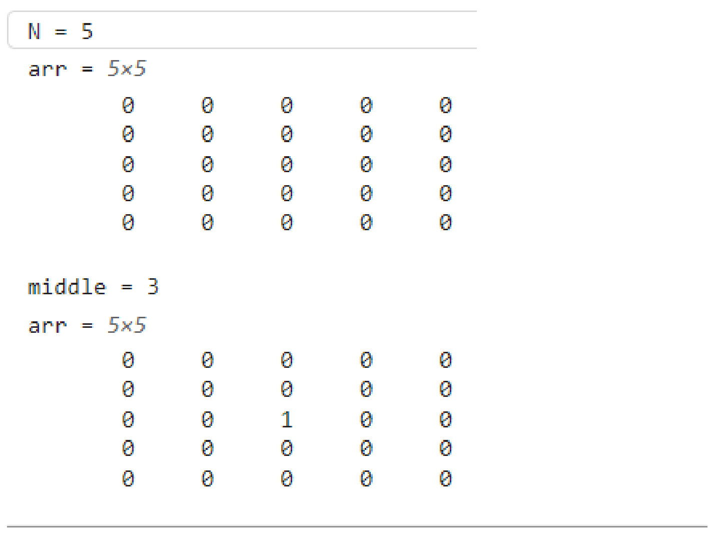
5.1(a) 🗎

```
N = 3

arr = zeros(N)
middle = ceil(N/2)
arr(middle, middle) = 1
```

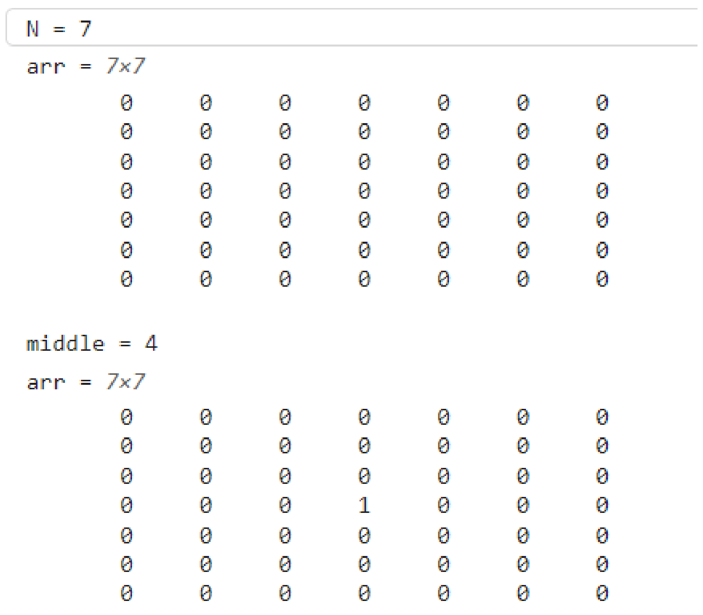
5.1(b) 🗎



```
N = 3
arr = 3x3
       0     0     0
       0     0     0
       0     0     0

middle = 2
arr = 3x3
       0     0     0
       0     1     0
       0     0     0
```

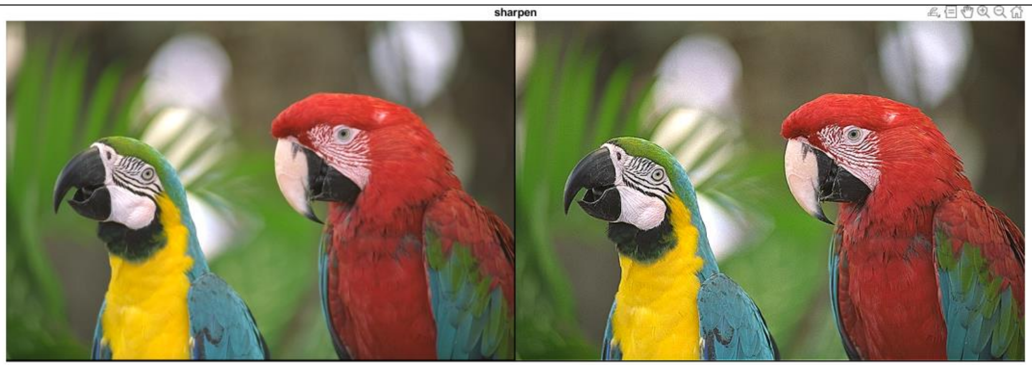


```
N = 5
arr = 5×5
        0     0     0     0     0
        0     0     0     0     0
        0     0     0     0     0
        0     0     0     0     0
        0     0     0     0     0

middle = 3
arr = 5×5
        0     0     0     0     0
        0     0     0     0     0
        0     0     1     0     0
        0     0     0     0     0
        0     0     0     0     0
```



```
N = 7
arr = 7×7
        0     0     0     0     0     0     0
        0     0     0     0     0     0     0
        0     0     0     0     0     0     0
        0     0     0     0     0     0     0
        0     0     0     0     0     0     0
        0     0     0     0     0     0     0
        0     0     0     0     0     0     0

middle = 4
arr = 7×7
        0     0     0     0     0     0     0
        0     0     0     0     0     0     0
        0     0     0     0     0     0     0
        0     0     0     1     0     0     0
        0     0     0     0     0     0     0
        0     0     0     0     0     0     0
        0     0     0     0     0     0     0
```

5.2	With every value except the center value being 0 in the array, the only data taken is the original pixel.
5.3	On the black border it is given RGB values of [0 0 0]
5.4	I see that the black border is mostly gone. With the average kernel active, we can see the black border bleed into blurry image of the parrots. But with the second line of code, absent of the ‘full’ switch, the image length and width reduce. Removing what would be the complete black border if it were unblurred.
5.5	The circular technique seems to create a wrap-around effect where a portion of the left appears on the right, a portion of the top appears on the bottom, and vice versa. In my opinion this does not work very well with the patriot picture, however this would work if the opposite edges of the picture where the same or if it worked in a artistic

	scope.
5.6 ✎	The replicate switch replaces the border with the pixel value of the same column if on the top edge and row if on the sides. Paired with the avg kernel, the replicate and replicate switch make the image appear to be one solid image that was blurred to the unknowing eye.
5.7 ✎	The symmetric command takes the side and does a sort of mirroring/ reflection on the edges. It gives the feeling that the parrots are standing on water only the water in all around them. I think symmetric is the best for the parrots and most other images that are taken. It feels more natural and fluid then the other commands that we went over.
5.8 ✎	 We chose the symmetrical switch with the sharpen kernel. We liked it because instead of making the image look sharper, it makes it look like it was drawn to be hyper realistic.
6.1 ✎	Fox – I am still learning how the program works and don’t feel great yet. However, I am getting to a good place. Luis – I feel very comfortable. I have yet to branch out beyond what we have done in class, but as I learn more and more syntax as well as tips and tricks as we go on with the quarter I fele increasingly comfortable to try something new on my own accord.