Introduction to Intelligent Systems - Lab 5

Diego Velasco Volkmann Eduardo Faccin Vernier
S2851059 S3012875

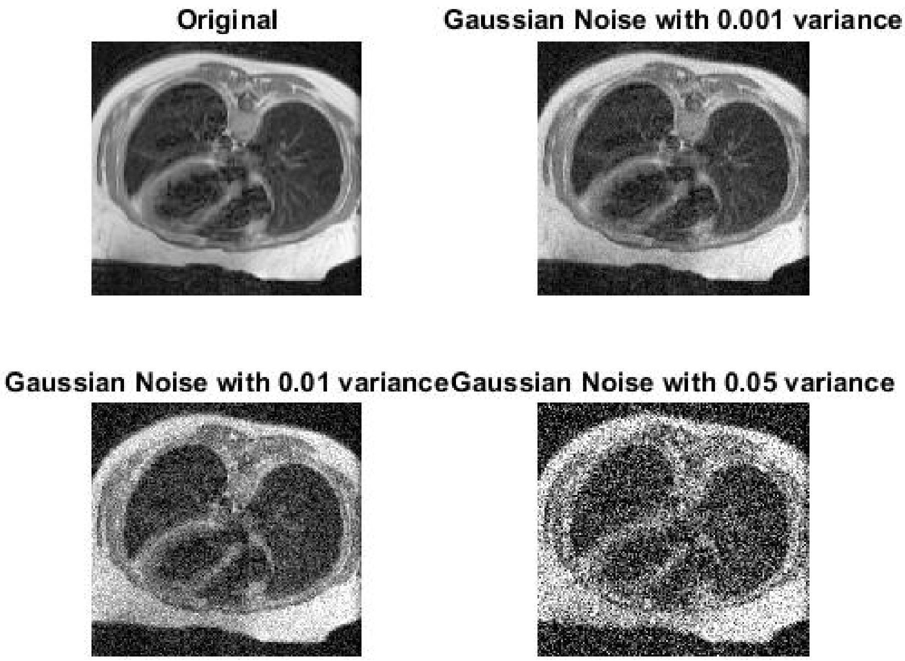
October 31, 2015

Exercise 1:

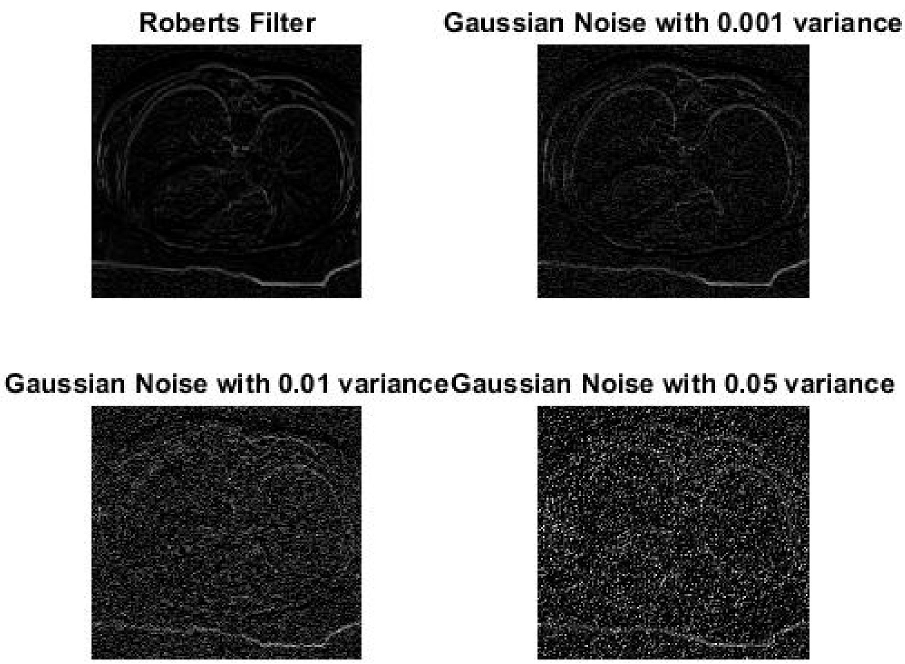
To perform this assignment, a convolution function was implemented. It takes an image and a 3x3 kernel as arguments and perform convolution on a padded image with repeated boundary pixel values.

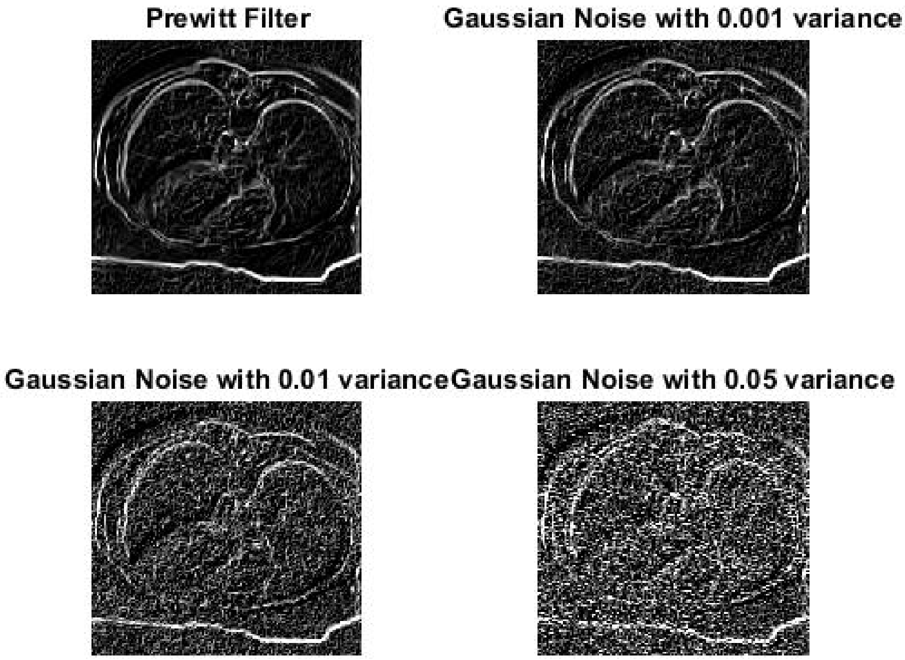
```
function resultImage = convolution( originalImage, kernel )
%CONVOLUTION takes an image and a 3x3 kernel and applies convolution

   resultImage = zeros(length(originalImage(:,1)),length(originalImage(1,:)));
   % padding image with boundary pixels
   paddedImage = padarray (originalImage, [1 1], 'replicate', 'both');

 % 180 degree rotation of kernel in order to perform convolution, not
     correlation
 % K   [0]      [1]      [2]        [3]      [4]     [5]      [6]       [7]       [8]
 %   (x+1,y+1)  (x,y+1)  (x-1,y+1)  (x+1,y)   (x,y)  (x-1,y)  (x+1,y-1)  (x,y-1)  (x-1,
     y-1)

   for x = 2:(length(paddedImage(:,1))-1)
       for y = 2:(length(paddedImage(1,:))-1)
           newValue = 0;
           newValue = newValue + (paddedImage(x+1, y+1)) * kernel (1);
           newValue = newValue + (paddedImage(x,   y+1)) * kernel (2);
           newValue = newValue + (paddedImage(x-1, y+1)) * kernel (3);
           newValue = newValue + (paddedImage(x+1,   y)) * kernel (4);
           newValue = newValue + (paddedImage(x,     y)) * kernel (5);
           newValue = newValue + (paddedImage(x-1,   y)) * kernel (6);
           newValue = newValue + (paddedImage(x+1, y-1)) * kernel (7);
           newValue = newValue + (paddedImage(x   ,y-1)) * kernel (8);
           newValue = newValue + (paddedImage(x-1, y-1)) * kernel (9);
           if (newValue < 0)
               newValue = 0;
           end
           if(newValue>1)
               newValue = 1;
           end
           resultImage(x-1,y-1) = newValue;
       end
    end
end
```

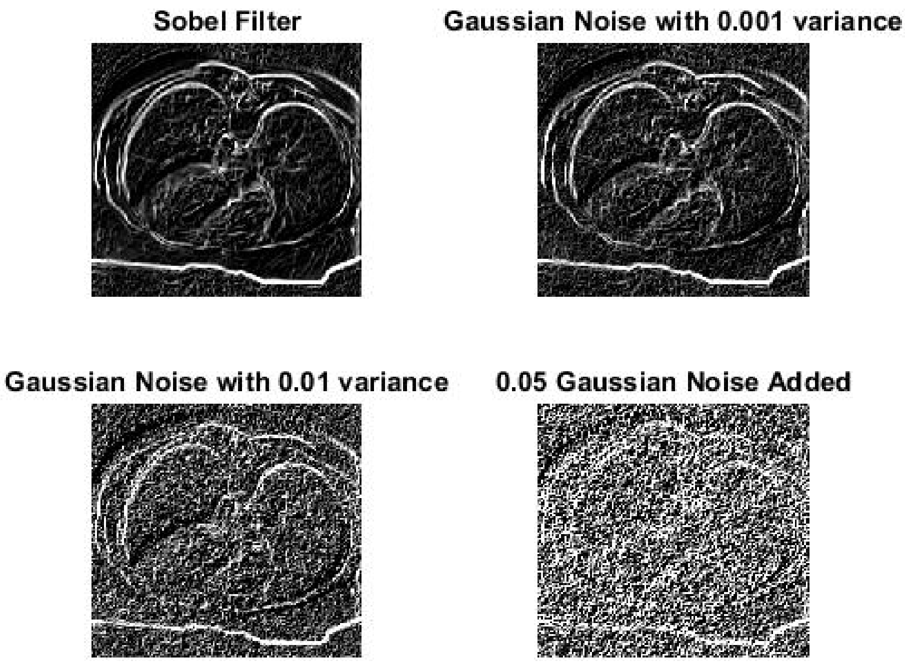
Each filter was applied in four images: the original chest image, and the chest image with 0.001, 0.01 and 0.05 Gaussian noise added.



The results of the filtering are displayed below for each filter kernel.



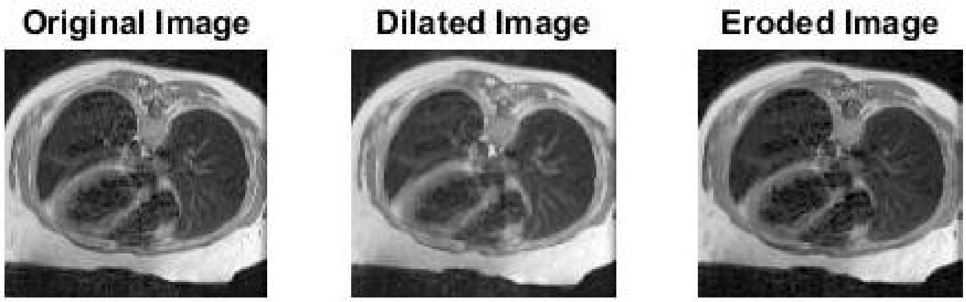




Noise has a big influence in all of theses filters considering they are all quite small (3x3 or 2x2). The smaller the filter, the more noise sensitive it is, therefore, as it is visible in the images above, Roberts Cross (2x2) convolutions get very degraded as the noise is increased compared to the other two convolution kernels. Increasing the size of the filters would make the filtering less noise, but, as a trade off, would result in a poorer localization of features.

Exercise 2:

First in the assignment, we implement the most basic morphological functions. In a binary image, as seen in class, when applying Erosion, if any pixel inside the structuring elements is has a value of 0 in the neighborhood, the output pixel is also set to 0. But dealing with grey scale images, the value of the output pixel is the minimum value of all the pixels in the input pixel's neighborhood. The same applies to Dilation, but instead of the minimum value, the output pixel is set as the maximum.



The morphological edge detector used in this assignment is Boundary Extraction. It consists on subtracting from the original A image an eroded version of it with a B Structuring Element.

$$\beta(A) = A - (A \ominus B)$$

The results of this technique with different Structuring Elements are displayed below.

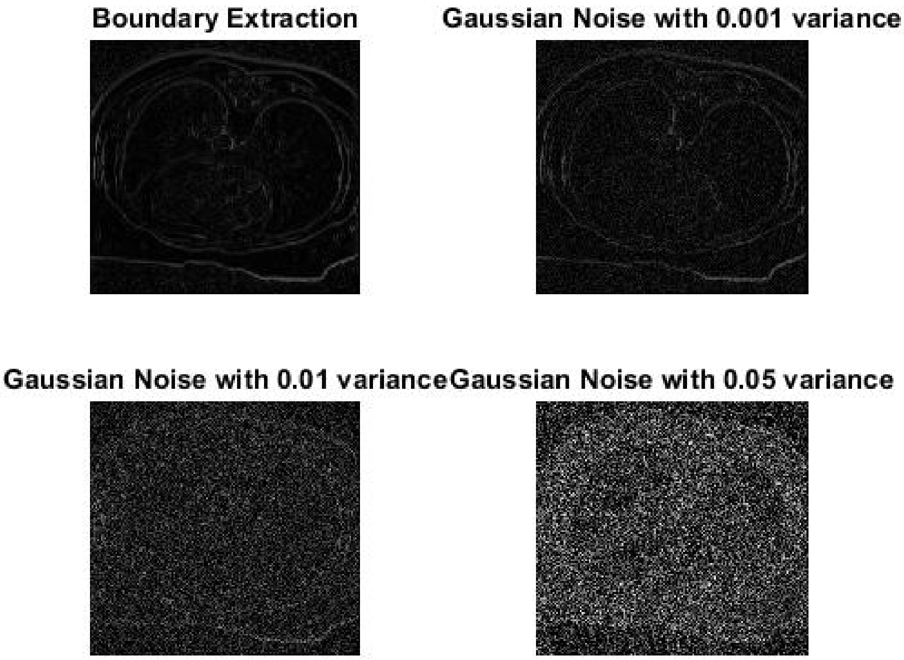


Figure 1: Structuring element = [0 1 0; 1 1 1; 0 1 0]

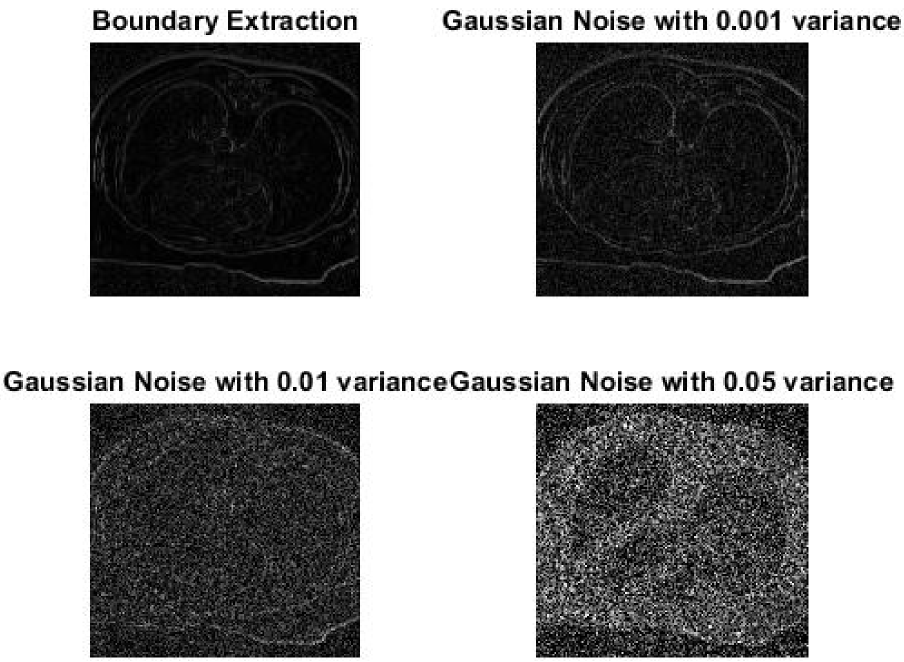


Figure 2: Structuring element = [1 1 1; 1 1 1; 1 1 1]

As all previous techniques, this Boundary Extraction is also very noise sensitive. But comparing the results obtained from each structuring element, it is safe to say that the [1 1 1; 1 1 1; 1 1 1] structuring element is slightly more robust to noise addition.

Exercise 3:

Modulating the parameters on the Canny Edge algorithm it is possible to generate a very noise resistant filtering. What our group observed is that increasing the value of the minimum threshold and sigma, most of the noise is discarded and only strong edges are highlighted. That is very clear comparing Figure 3, which has low threshold and sigma, to Figure 6, for example. But this increase in the parameters comes at a cost - edges that are not very strong, but might be important for a diagnoses, for instance, will be disregarded. Increasing the Threshold to very high numbers, as in Figure 7 will also result in edges loss. For noisy images, we've come to the conclusion that high sigma values (as srqt(4) and sqrt(8)) with threshold values of [0.1 0.25] are a good trade off between noise resistance and edge strength. Compared to the previous techniques, in this application, Canny seems to be the best by a large margin.

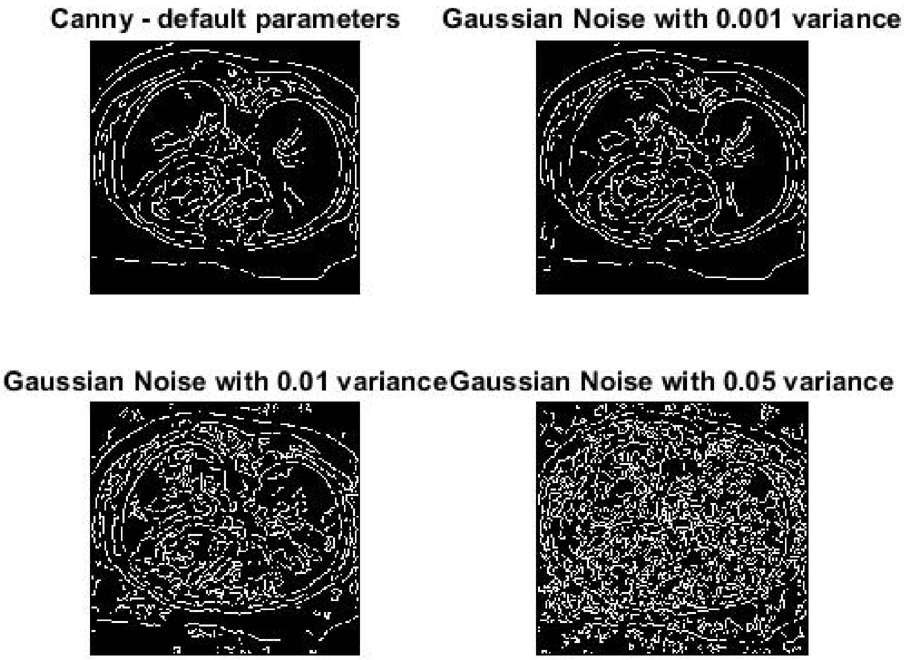


Figure 3: Default values (Threshold chosen automatically by MATLAB and sigma = sqrt(2))

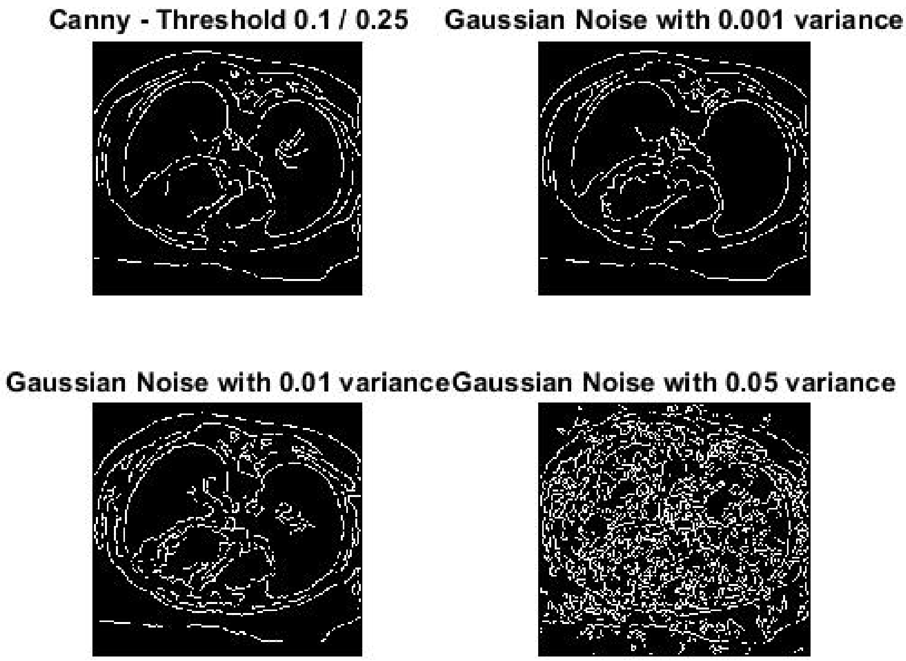


Figure 4: Threshold is set as [0.1 0.25] and sigma is default = sqrt(2)

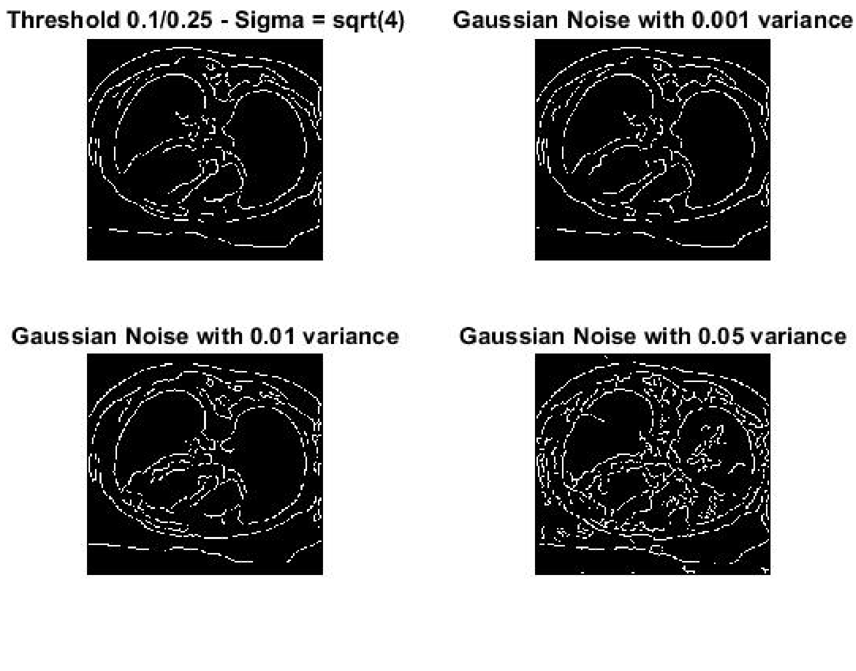


Figure 5: Threshold is set as [0.1 0.25] and sigma = sqrt(4)

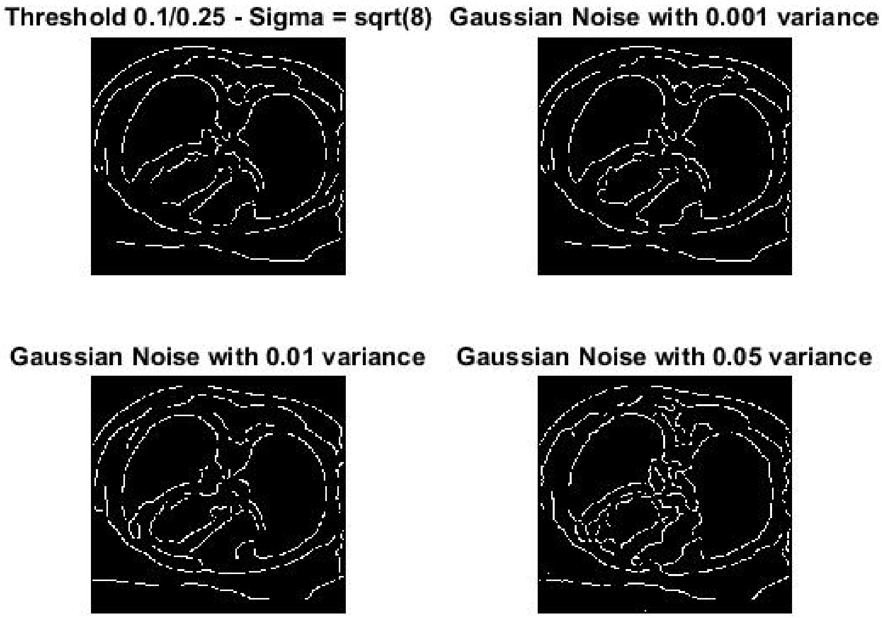


Figure 6: Threshold is set as [0.1 0.25] and sigma = sqrt(8)

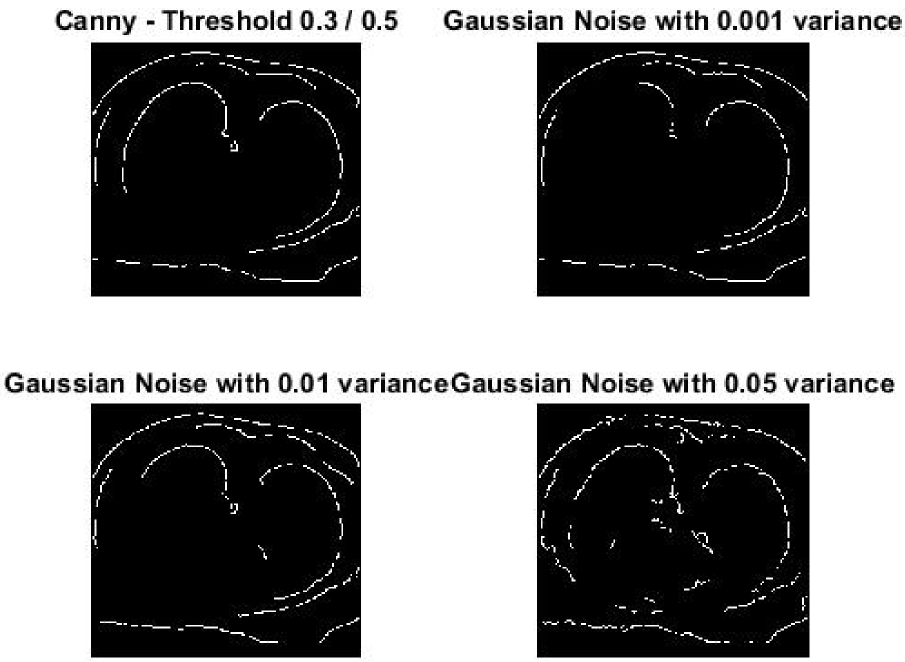


Figure 7: Threshold is set very high [0.3 0.5] and sigma is default = sqrt(2)

Exercise 4:

Study the paper Contour and boundary detection improved by surround suppression of texture edges by Grigorescu, Petkov, and Westenberg provided on Nestor. This is the predecessor of the methods described in the lecture. The method can be accessed through a web implementation found at:

$$http://www.cs.rug.nl/~imaging$$

Click on Canny edge detector (Canny filter) for image processing and computer vision:, which will lead you to a web application which can perform both the Canny detector and two surround inhibition methods. Select the kanizsa.png image as the target and update the view. Try various settings of surround inhibition, and explain the differences in output. Do the same for popout.png. What is the difference with kanizsa.png? Hint: Set the sigma (around 4 seems to work) and K2 (around 8) parameters high enough to get the expected inhibition effects.
Answer: According to Figure 10, when there's too much surrounding inhibition, popout.png becomes ”invisible”, unlike kanizsa.png, shown on Figure 11, which survives with a few elements, since popout is composed of several short lines, but kanisza has a full square outline. The surround inhibition is meant to suppress texture edges while leaving relatively unaffected the contours of objects and region boundaries, therefore removing short and non-continuous edges. The value of sigma changes the edge outline to a more rounded shape, depending on how high sigma is.

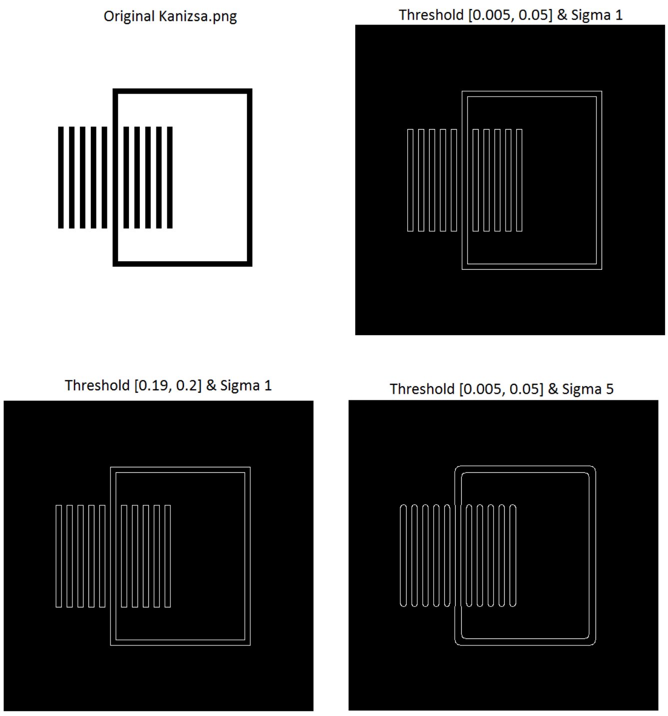


Figure 8: Kanizsa.png tests

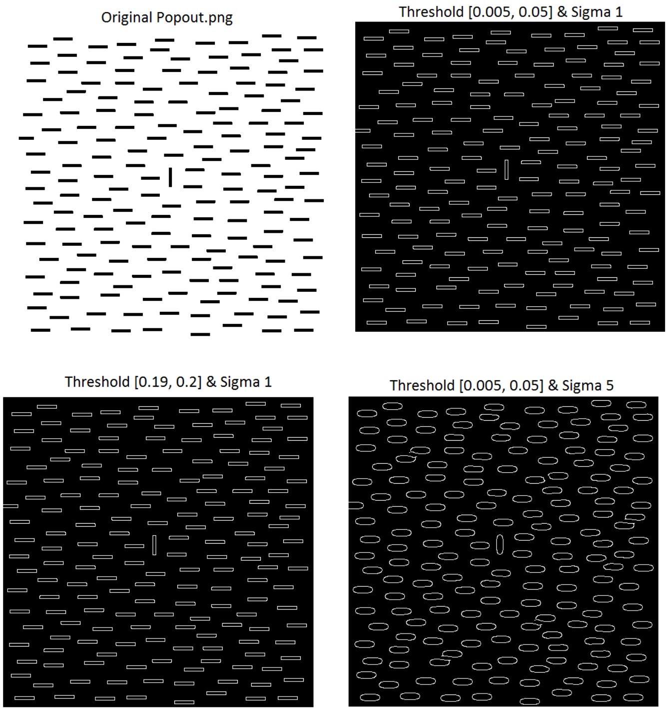


Figure 9: Popout.png tests

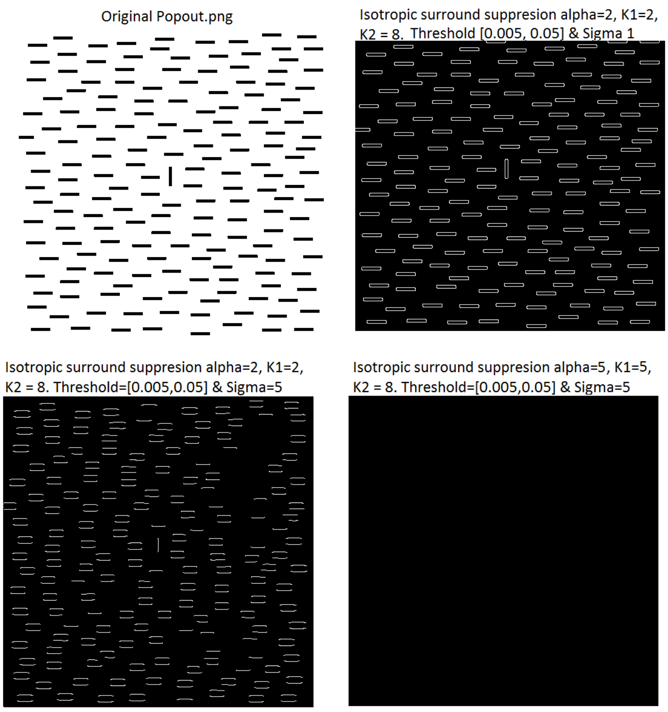


Figure 10: Kanizsa.png tests with surround inhibition

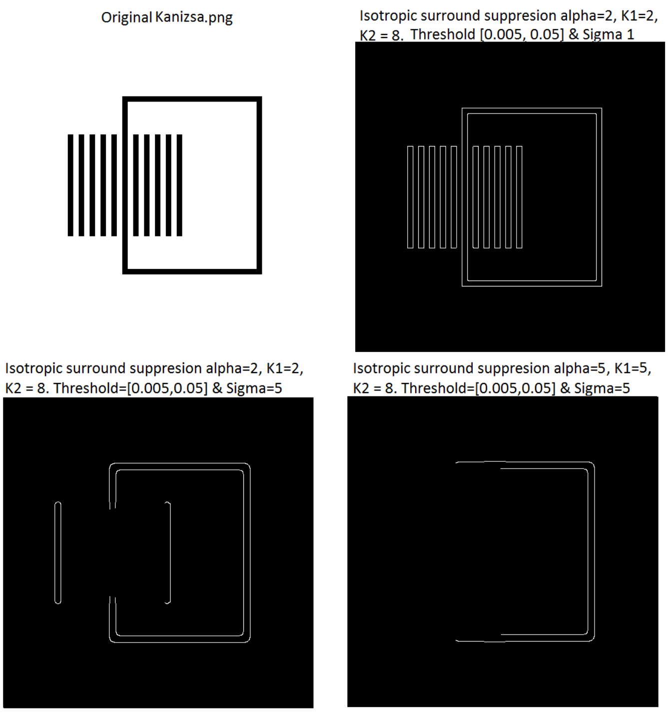


Figure 11: Popout.png tests with surround inhibition

Division of work

Eduardo was responsible for exercise 1 and 2, while Diego was responsible for exercise 3 and 4. Later on, Eduardo deleted Diego's answer on exercise 3, writing a new answer.

Appendix:

Listing 1: lab5q1.m file - function calls and plots using the convolution function shown in the first exercise

```
close all;

I = im2double(imread('chest.pgm'));

% Convolution Kernels
hRbt1 = [0 0 0; 0 -1 0; 0 0 1];
hRbt2 = [0 0 0; 0 0 -1; 0 1 0];
hPwt1 = [-1 -1 -1; 0 0 0; 1 1 1];
hPwt2 = [-1 0 1; -1 0 1; -1 0 1];
hSbl1 = [-1 -2 -1; 0 0 0; 1 2 1];
hSbl2 = [-1 0 1; -2 0 2; -1 0 1];

% Images with added noise
I0001 = imnoise(I, 'gaussian', 0, 0.001);
I001 = imnoise(I, 'gaussian', 0, 0.01);
I005 = imnoise(I, 'gaussian', 0, 0.05);

% Images with different amounts of noise
figure;
subplot(2, 2, 1);
imshow(I);
title('Original');
subplot(2, 2, 2);
imshow(I0001);
title('Gaussian Noise with 0.001 variance');
subplot(2, 2, 3);
imshow(I001);
title('Gaussian Noise with 0.01 variance');
subplot(2, 2, 4);
imshow(I005);
title('Gaussian Noise with 0.05 variance');

% Roberts kernel
figure;
Rbt1 = convolution(I, hRbt1);
Rbt2 = convolution(I, hRbt2);
Rbt = sqrt(Rbt1.^2+Rbt2.^2);
subplot(2, 2, 1);
imshow(Rbt);
title('Roberts Filter');

Rbt1 = convolution(I0001, hRbt1);
Rbt2 = convolution(I0001, hRbt2);
Rbt = sqrt(Rbt1.^2+Rbt2.^2);
subplot(2, 2, 2);
imshow(Rbt);
title('Gaussian Noise with 0.001 variance');

Rbt1 = convolution(I001, hRbt1);
Rbt2 = convolution(I001, hRbt2);
Rbt = sqrt(Rbt1.^2+Rbt2.^2);
subplot(2, 2, 3);
imshow(Rbt);
title('Gaussian Noise with 0.01 variance');

Rbt1 = convolution(I005, hRbt1);
Rbt2 = convolution(I, hRbt2);
Rbt = sqrt(Rbt1.^2+Rbt2.^2);
subplot(2, 2, 4);
imshow(Rbt);
title('Gaussian Noise with 0.05 variance');


% Prewitt kernel
figure;
Pwt1 = convolution(I, hPwt1);
Pwt2 = convolution(I, hPwt2);
Pwt = sqrt(Pwt1.^2+Pwt2.^2);
subplot(2, 2, 1);
imshow(Pwt);
title('Prewitt Filter');

Pwt1 = convolution(I0001, hPwt1);
Pwt2 = convolution(I0001, hPwt2);
Pwt = sqrt(Pwt1.^2+Pwt2.^2);
```

```
subplot(2, 2, 2);
imshow(Pwt);
title('Gaussian Noise with 0.001 variance');

Pwt1 = convolution(I001, hPwt1);
Pwt2 = convolution(I001, hPwt2);
Pwt = sqrt(Pwt1.^2+Pwt2.^2);
subplot(2, 2, 3);
imshow(Pwt);
title('Gaussian Noise with 0.01 variance');

Pwt1 = convolution(I005, hPwt1);
Pwt2 = convolution(I001, hPwt2);
Pwt = sqrt(Pwt1.^2+Pwt2.^2);
subplot(2, 2, 4);
imshow(Pwt);
title('Gaussian Noise with 0.05 variance');


% Sobel kernel vertical
figure;
Sbl1 = convolution(I, hSbl1);
Sbl2 = convolution(I, hSbl2);
Sbl = sqrt(Sbl1.^2+Sbl2.^2);
subplot(2, 2, 1);
imshow(Sbl);
title('Sobel Filter');

Sbl1 = convolution(I0001, hSbl1);
Sbl2 = convolution(I0001, hSbl2);
Sbl = sqrt(Sbl1.^2+Sbl2.^2);
subplot(2, 2, 2);
imshow(Sbl);
title('Gaussian Noise with 0.001 variance');

Sbl1 = convolution(I001, hSbl1);
Sbl2 = convolution(I001, hSbl2);
Sbl = sqrt(Sbl1.^2+Sbl2.^2);
subplot(2, 2, 3);
imshow(Sbl);
title('Gaussian Noise with 0.01 variance');

Sbl1 = convolution(I005, hSbl1);
Sbl2 = convolution(I005, hSbl2);
Sbl = sqrt(Sbl1.^2+Sbl2.^2);
subplot(2, 2, 4);
imshow(Sbl);
title('0.05 Gaussian Noise Added');
```

Listing 2: lab5q2.m file - function calls and plots from the Boundary Extraction method. Erosion and Dilation functions are listed below.

```
close all;

I = im2double(imread('chest.pgm'));
I0001 = imnoise(I, 'gaussian', 0, 0.001);
I001 = imnoise(I, 'gaussian', 0, 0.01);
I005 = imnoise(I, 'gaussian', 0, 0.05);
% Structuring element
SE1 = [ 0 1 0; 1 1 1; 0 1 0];
SE2 = [ 1 1 1; 1 1 1; 1 1 1];

dilatedImage = dilation (I, SE1);
erodedImage = erosion (I, SE1);

figure;
subplot (1,3,1);
imshow(I);
title ('Original Image');
subplot(1,3,2);
imshow(dilatedImage);
title('Dilated Image');
subplot(1,3,3);
imshow(erodedImage);
title('Eroded Image');



erodedImage0001 = erosion (I0001, SE1);
```

```
erodedImage001 = erosion (I001, SE1);
erodedImage005 = erosion (I005, SE1);


figure;
boundaryExtraction = I - erodedImage;
subplot(2,2,1);
imshow(boundaryExtraction);
title('Boundary Extraction');

boundaryExtraction = I0001 - erodedImage0001;
subplot(2,2,2);
imshow(boundaryExtraction);
title('Gaussian Noise with 0.001 variance');

boundaryExtraction = I001 - erodedImage001;
subplot(2,2,3);
imshow(boundaryExtraction);
title('Gaussian Noise with 0.01 variance');


boundaryExtraction = I005 - erodedImage005;
subplot(2,2,4);
imshow(boundaryExtraction);
title('Gaussian Noise with 0.05 variance');


% Using different SE
erodedImage0001 = erosion (I0001, SE2);
erodedImage001 = erosion (I001, SE2);
erodedImage005 = erosion (I005, SE2);


figure;
boundaryExtraction = I - erodedImage;
subplot(2,2,1);
imshow(boundaryExtraction);
title('Boundary Extraction');

boundaryExtraction = I0001 - erodedImage0001;
subplot(2,2,2);
imshow(boundaryExtraction);
title('Gaussian Noise with 0.001 variance');

boundaryExtraction = I001 - erodedImage001;
subplot(2,2,3);
imshow(boundaryExtraction);
title('Gaussian Noise with 0.01 variance');


boundaryExtraction = I005 - erodedImage005;
subplot(2,2,4);
imshow(boundaryExtraction);
title('Gaussian Noise with 0.05 variance');
```

Listing 3: dilation.m file

```
function  resultImage  = dilation( I, SE )
    resultImage = zeros(length(I(:,1)), length(I(1,:)));

    for(i = 1:length(I(:,1)))
        for(j = 1:length(I(1,:)))
            [Y X] = meshgrid (-1:1, -1:1);
            X = SE.*(X + i);
            Y = SE.*(Y + j);

            indexArray(:,:,1)= X;
            indexArray(:,:,2)= Y;

            max = -1;
            for k = 1:3
                for l = 1:3
                    if ((indexArray(k,l,1)>0 && indexArray(k,l,1) < length (I(:,1))) && ...
                        (indexArray(k,l,2)>0 && indexArray(k,l,2) < length (I(1,:))))
                        if (I(indexArray(k,l,1),indexArray(k,l,2)) > max)
                            max = I(indexArray(k,l,1),indexArray(k,l,2));
                        end
                    end
                end
```

```
            end
            resultImage(i,j) = max;
        end
    end
end
```

Listing 4: erosion.m file

```
function  resultImage  = erosion( I, SE )
    resultImage = zeros(length(I(:,1)), length(I(1,:)));

    for(i = 1:length(I(:,1)))
        for(j = 1:length(I(1,:)))
            [Y X] = meshgrid (-1:1, -1:1);
            X = SE.*(X + i);
            Y = SE.*(Y + j);

            indexArray(:,:,1)= X;
            indexArray(:,:,2)= Y;

            min = 1;
            for k = 1:3
                for l = 1:3
                    if ((indexArray(k,l,1)>0 && indexArray(k,l,1) < length (I(:,1))) && ...
                        (indexArray(k,l,2)>0 && indexArray(k,l,2) < length (I(1,:))))
                        if (I(indexArray(k,l,1),indexArray(k,l,2)) < min)
                            min = I(indexArray(k,l,1),indexArray(k,l,2));
                        end
                    end
                end
            end
            resultImage(i,j) = min;
        end
    end
end
```

Listing 5: lab5q3.m file - function calls and plots using for the Canny Edge technique

```
close all;

I = im2double(imread('chest.pgm'));
I0001 = imnoise(I, 'gaussian', 0, 0.001);
I001 = imnoise(I, 'gaussian', 0, 0.01);
I005 = imnoise(I, 'gaussian', 0, 0.05);

figure;
subplot(2, 2, 1);
BW = edge(I, 'Canny');
imshow(BW);
title('Canny - default parameters');

subplot(2, 2, 2);
BW = edge(I0001, 'Canny');
imshow(BW);
title('Gaussian Noise with 0.001 variance');

subplot(2, 2, 3);
BW = edge(I001, 'Canny');
imshow(BW);
title('Gaussian Noise with 0.01 variance');

subplot(2, 2, 4);
BW = edge(I005, 'Canny');
imshow(BW);
title('Gaussian Noise with 0.05 variance');

%%%%%%%%
figure;
subplot(2, 2, 1);
BW = edge(I, 'Canny', [0.1 0.25]);
imshow(BW);
title('Canny - Threshold 0.1 / 0.25');

subplot(2, 2, 2);
BW = edge(I0001, 'Canny', [0.1 0.25]);
imshow(BW);
title('Gaussian Noise with 0.001 variance');

```

```
subplot(2, 2, 3);
BW = edge(I001, 'Canny', [0.1 0.25]);
imshow(BW);
title('Gaussian Noise with 0.01 variance');

subplot(2, 2, 4);
BW = edge(I005, 'Canny', [0.1 0.25]);
imshow(BW);
title('Gaussian Noise with 0.05 variance');

%%%%%%%%
figure;
subplot(2, 2, 1);
BW = edge(I, 'Canny', [0.1 0.25], sqrt(4));
imshow(BW);
title('Threshold 0.1/0.25 - Sigma = sqrt(4)');

subplot(2, 2, 2);
BW = edge(I0001, 'Canny', [0.1 0.25], sqrt(4));
imshow(BW);
title('Gaussian Noise with 0.001 variance');

subplot(2, 2, 3);
BW = edge(I001, 'Canny', [0.1 0.25], sqrt(4));
imshow(BW);
title('Gaussian Noise with 0.01 variance');

subplot(2, 2, 4);
BW = edge(I005, 'Canny', [0.1 0.25], sqrt(4));
imshow(BW);
title('Gaussian Noise with 0.05 variance');
%%%%%%%%
figure;
subplot(2, 2, 1);
BW = edge(I, 'Canny', [0.1 0.25], sqrt(8));
imshow(BW);
title('Threshold 0.1/0.25 - Sigma = sqrt(8)');

subplot(2, 2, 2);
BW = edge(I0001, 'Canny', [0.1 0.25], sqrt(8));
imshow(BW);
title('Gaussian Noise with 0.001 variance');

subplot(2, 2, 3);
BW = edge(I001, 'Canny', [0.1 0.25], sqrt(8));
imshow(BW);
title('Gaussian Noise with 0.01 variance');

subplot(2, 2, 4);
BW = edge(I005, 'Canny', [0.1 0.25], sqrt(8));
imshow(BW);
title('Gaussian Noise with 0.05 variance');

%%%%%%%%
figure;
subplot(2, 2, 1);
BW = edge(I, 'Canny', [0.3 0.5]);
imshow(BW);
title('Canny - Threshold 0.3 / 0.5');

subplot(2, 2, 2);
BW = edge(I0001, 'Canny', [0.3 0.5]);
imshow(BW);
title('Gaussian Noise with 0.001 variance');

subplot(2, 2, 3);
BW = edge(I001, 'Canny', [0.3 0.5]);
imshow(BW);
title('Gaussian Noise with 0.01 variance');

subplot(2, 2, 4);
BW = edge(I005, 'Canny', [0.3 0.5]);
imshow(BW);
title('Gaussian Noise with 0.05 variance');
```