CSE 691: Image and Video Processing

Spring 2020 Assignment 2

Edge Detection

Lichen Liang

02/15/2020

Objectives

- Understand the Canny edge detection method
- Understand each step of the algorithm and what does it do.
- Apply the edge detection to different images and see how it detects and process the image.
- Notice the effects of variables.

Method

To run the code, run ***main.m*** and make sure all function files are in same folder.

The output image is labeled as :

Out [Original Image Number][Its related corresponding matrix]

For example, out2Es represents it used image2, the Es matrix for image2.

1. First, the image is loaded and filtered with gaussian filtering. I did not use the Gaussian filter function from assignment one for two reasons. As I have mentioned in my previous lab report, my Gaussian filter function produces a similar result as MATLAB built-in function *imgaussfilt()* only when the sigma value is small. Since the gaussian filter size in my function is user-defined, whereas the filter size varies depending on the sigma value in the *imgaussfilt(),* when a large sigma value is picked, two functions will produce different results. The second reason is that to use my function, the image must be changed to double using *im2double()* for the filtered image to be displayed. Consequently, the grayscale is from 0 to 1 rather than 0 to 255.
   In this assignment, I used *imgaussfilt()* (proved by the professor). The filtered image still cannot be displayed unless the image is changed to double, which will change the scale. using *imtool(),* it would show a white image, but the values are still from 0 to 255.

2. For Canny Enhancement, I created a horizontal filter $[-1 \quad 0 \quad 1]$ and a vertical filter $\begin{bmatrix}-1\\0\\1\end{bmatrix}$. Next, convolve the horizontal filter with image to get Jx and vertical filter with image to get Jy. The Edge Strength matrix Es is then calculated from $E_s = \sqrt{J_x^2 + J_y^2}$. The Edge Orientation of the Normal matrix Eo is calculated from arctan $(\frac{J_y}{J_x})$. The result from arctangent in MATLAB is in radians, so multiply it by $\frac{180}{\pi}$ to get the angle in degrees. Make sure all angles are positive values by adding 180 to a negative value. Lastly, categorize each angle to its nearest region: 0, 45, 90 ,135.

3. The non-maximum suppression uses Es and Eo as input. For each Es, find its corresponding Eo and compare the two values along its Eo. Create a new matrix I_n where if Es has a value less than its neighbors along Eo, then set I_n to 0, otherwise set I_n to Es. I_n is initially zero padded and the algorithm ignores the border values as they do not have enough neighbors to be compared to.

4. The I_n from non-maximum suppression is then used for hysteresis thresholding. We need two matrix here, one for output and one to keep track of visited coordinates. Both matrices are zero padded. First, find a coordinate where $I_n > High\, Threshold\ T_h$, set the output of this location at 255 and mark as visited. Then use its Eo to compare its neighbors. If its neighbors have an In value greater than Low Threshold T_l, then this neighbor becomes our new center pixel. Use this new Eo to compare to this center pixel's neighbors. Repeat the process until no neighbors have a value larger then T_l or the coordinate has already been visited. Then look for next I_n larger than T_h.
   In the case where both neighbors are greater than T_l and we must choose one to move forward, I specified the one in the order of my code is written. For example, when angle is 90, I picked the bottom neighbor over the top simply because the bottom neighbor comes first in the code and nothing else.

Results and Discussion

In this part, I am going to primarily use the 'Syracuse_01.jpg' for discussion. I will use 'Flowers.jpg' to show the effects of Tl and 'Syracuse_02.jpg' to double check my results.

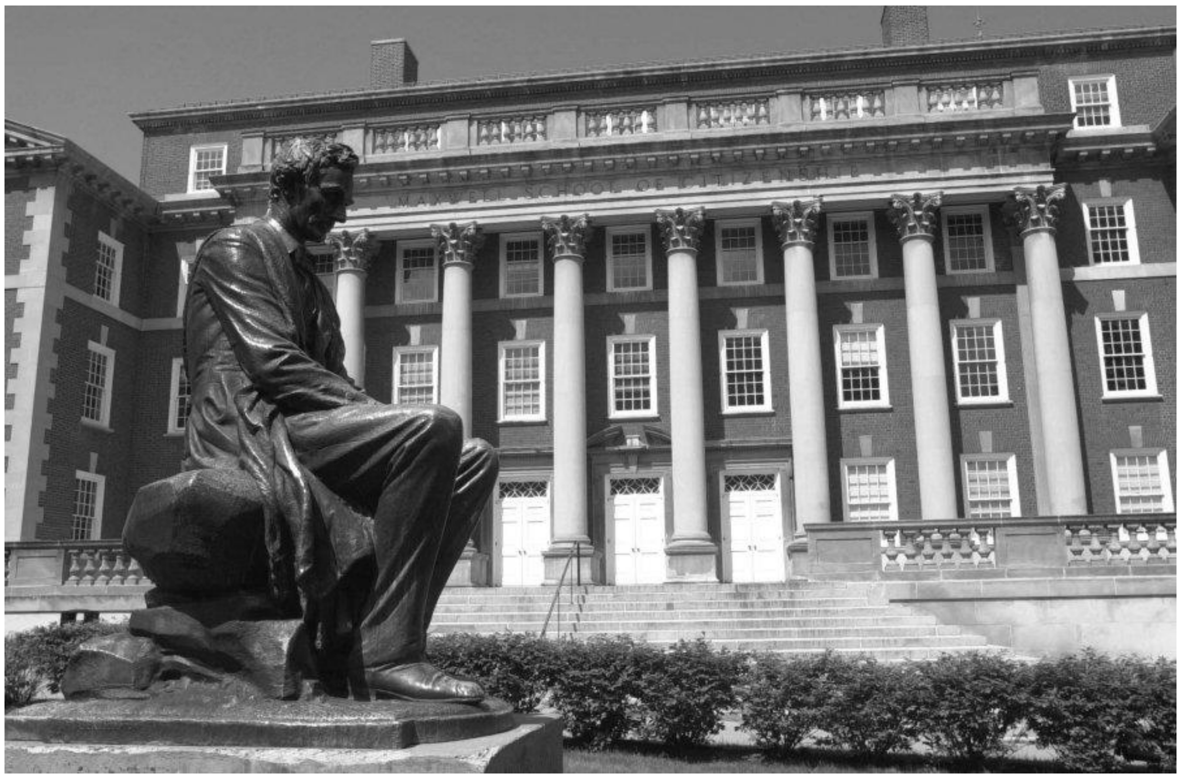
Figure 1. Original image of ‘Syracuse_01.jpg’

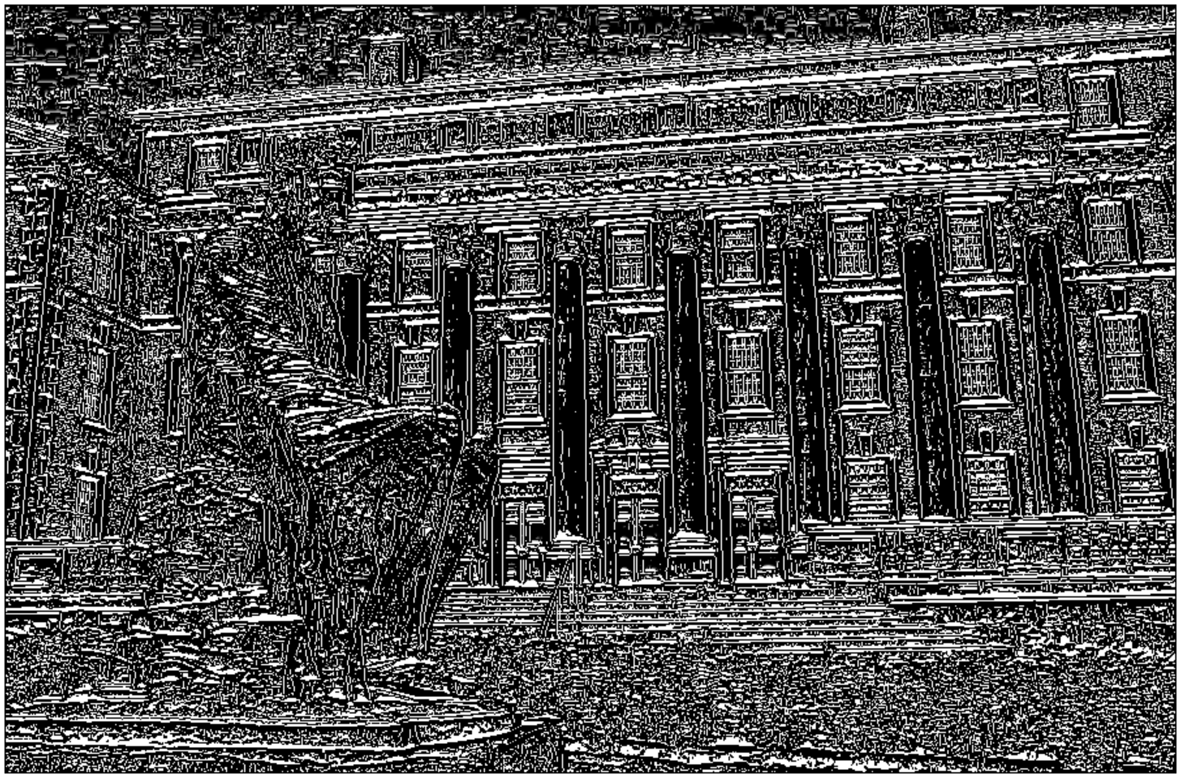
Figure 2. I_n (from non-max suppression) of Syracuse_01.jpg with Gaussian Sigma = 1

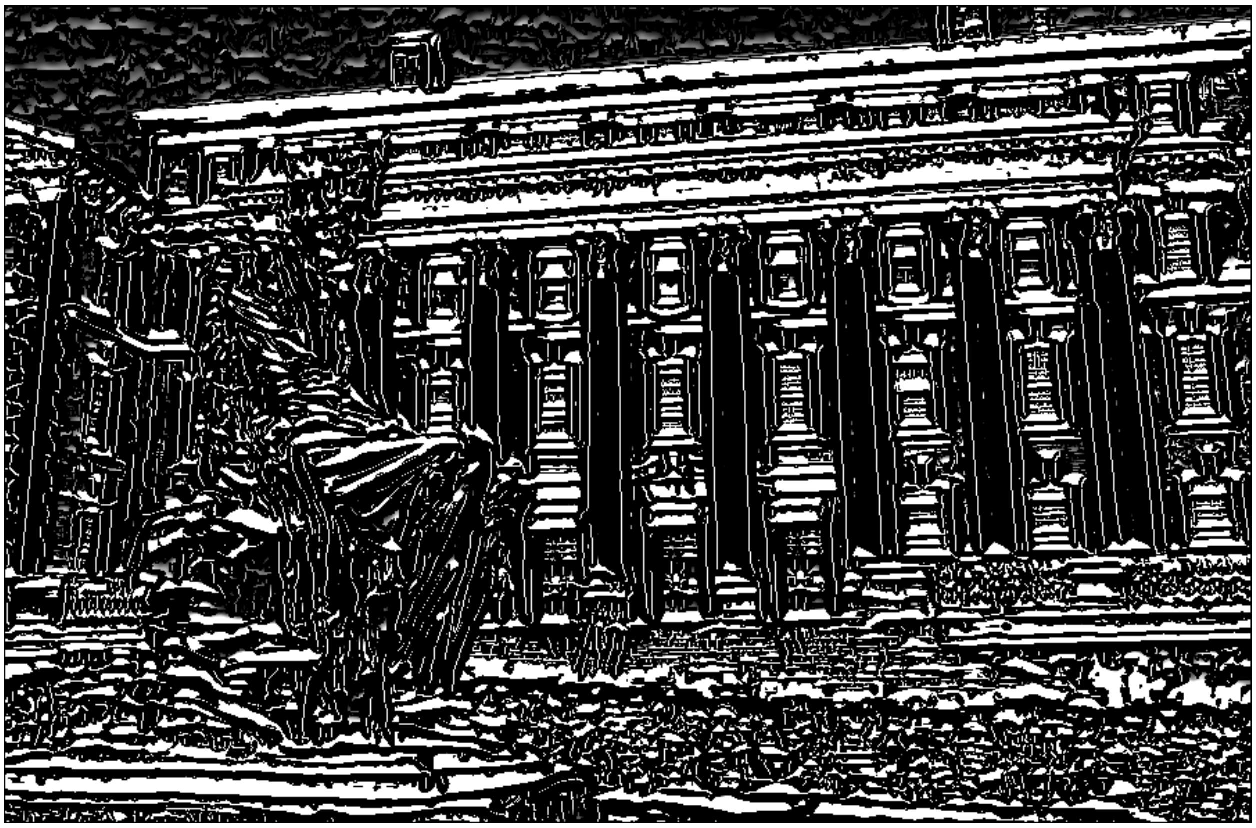
Figure 3. I_n of Syracuse_01.jpg with Gaussian Sigma = 3

Comparing figure 2 and figure 3, we can see that when we use an larger sigma value and larger filter size, the image becomes very blurry. Therefore, it is much harder to do non-max suppression because the edges are too wide. Figure 2 has the thins the edges in great details, but figure 3 shows that non-max suppression cannot thin the edges due to the too smoothed image.

Sigma of value one will be used for further processing.

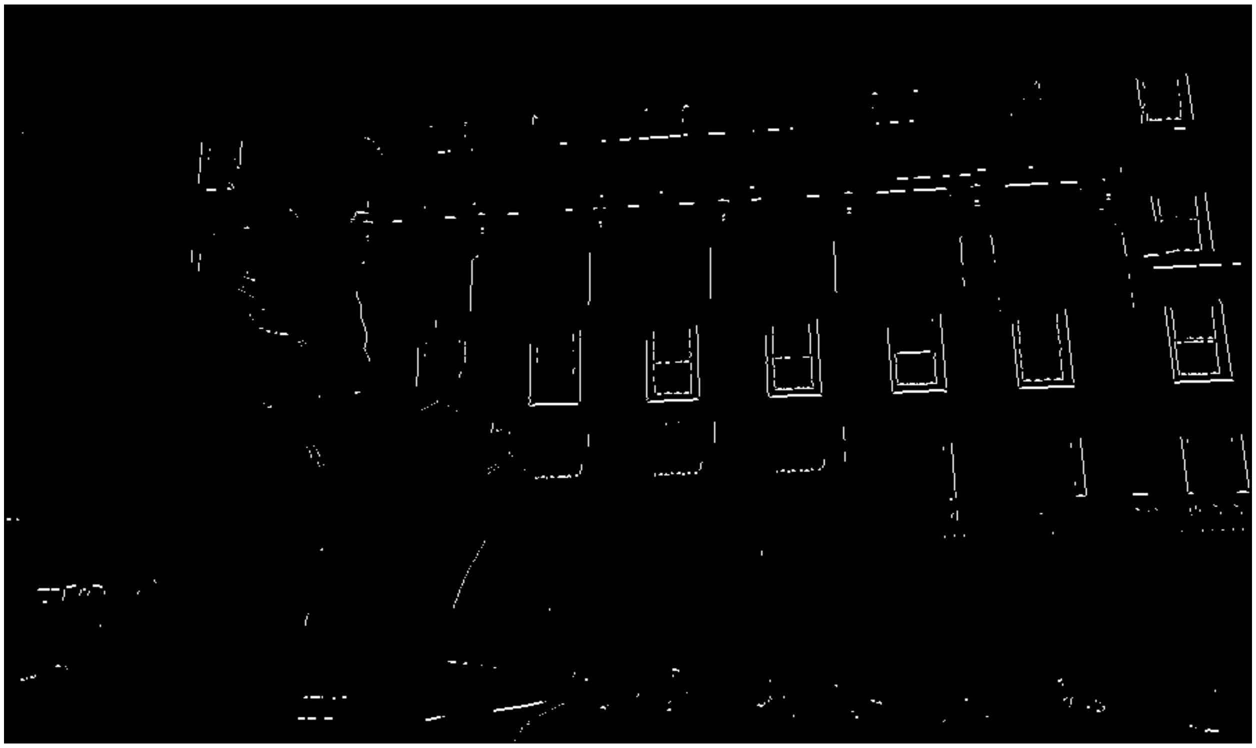

Figure 4. Out2HT(from hysteresis threshold) with Th = 80, Tl = 30

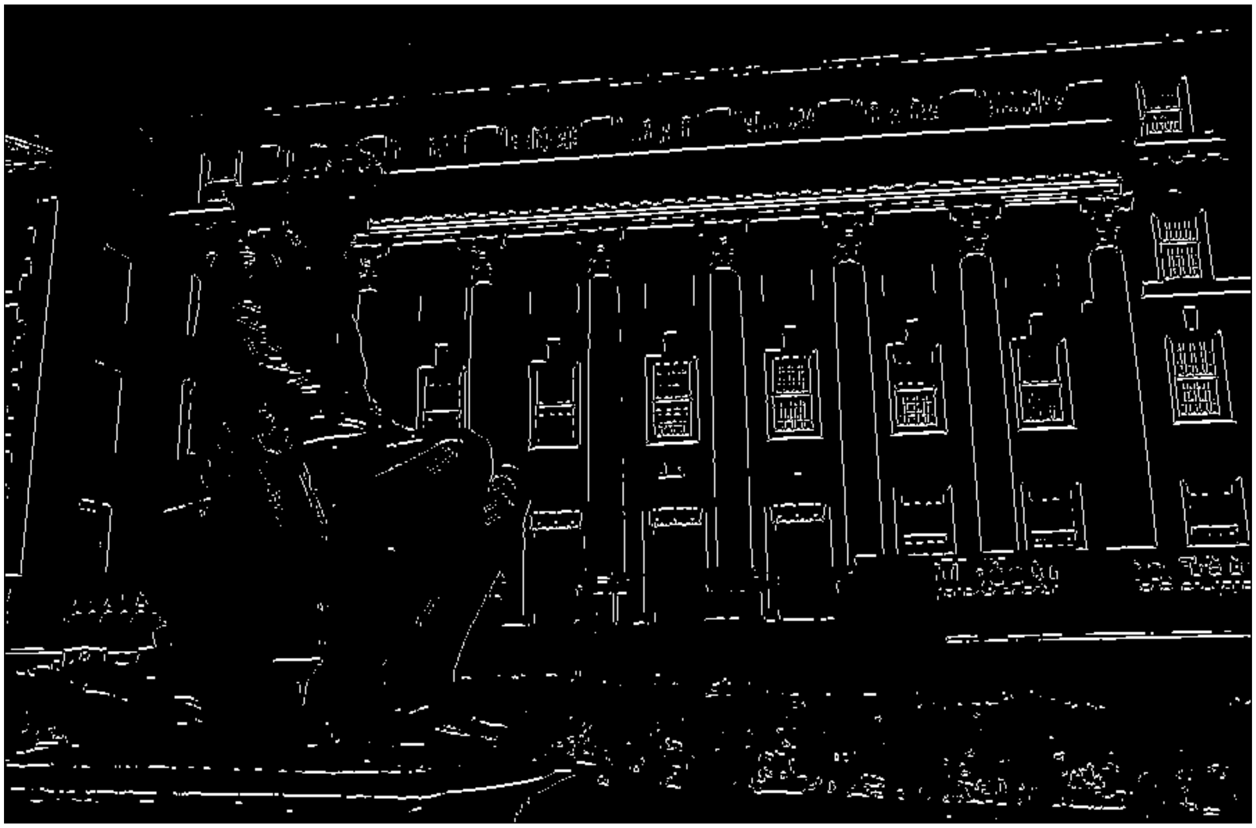

Figure 5. Out2HT(from hysteresis threshold) with Th = 50, Tl = 30

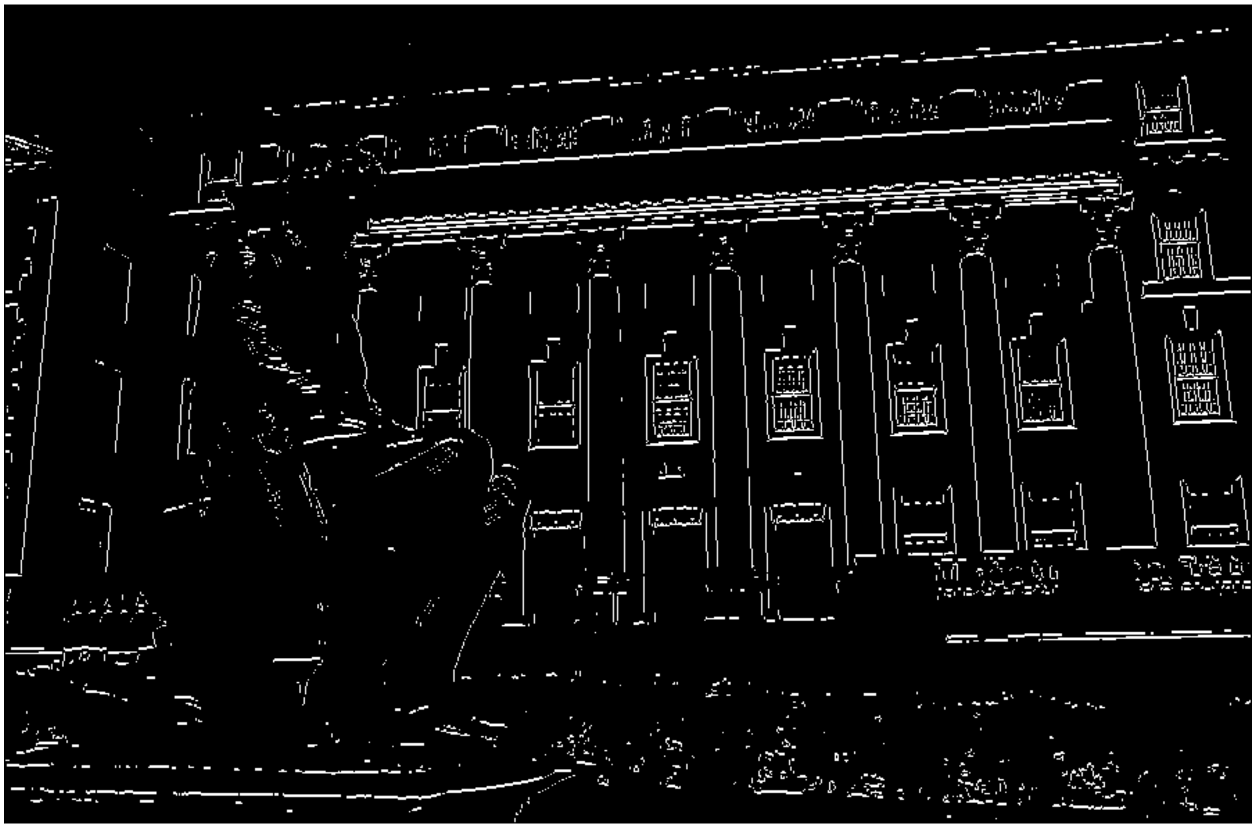

Figure 6. Out2HT(from hysteresis threshold) with Th = 50, Tl = 10

Comparing figures 4, 5, and 6, we can see the effects of different values for high and low thresholds. From figure 4 and 5, the higher the Th, the less edges in the output. When we pick the Th to be large, there are fewer starting points(with values larger than Th) to construct the output image. From figure 5 and 6, the lower the Tl, the more details that are considered to be an edge in the output image. It might not be visible from looking at figures 5 and 6, but there is a minor difference between different Tl values. I will explain this further in the next set of images.

Overall, the values that are larger than Th will be considered an strong edge. The values lower than Tl are set to 0. The values in between are considered an edge only if they have a connectivity to the strong edge in the direction of Eo. Otherwise, they will be set to 0 as well.

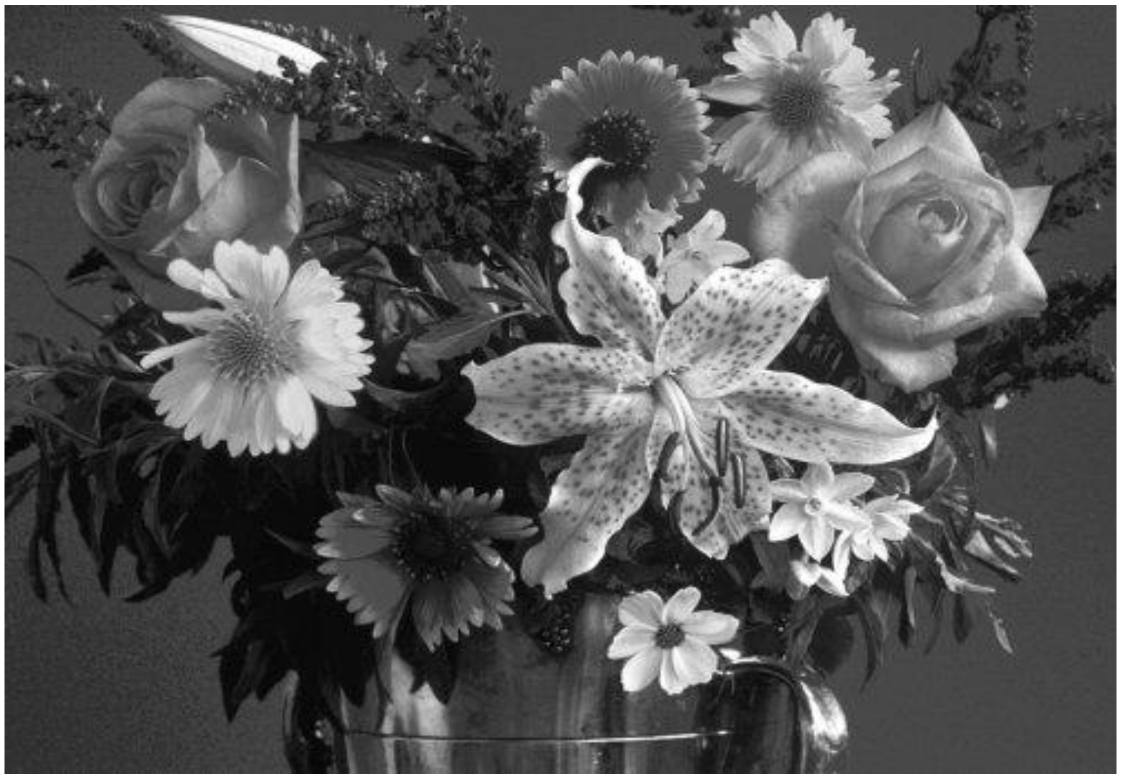
Figure 7. Original image of ‘Flowers.jpg’

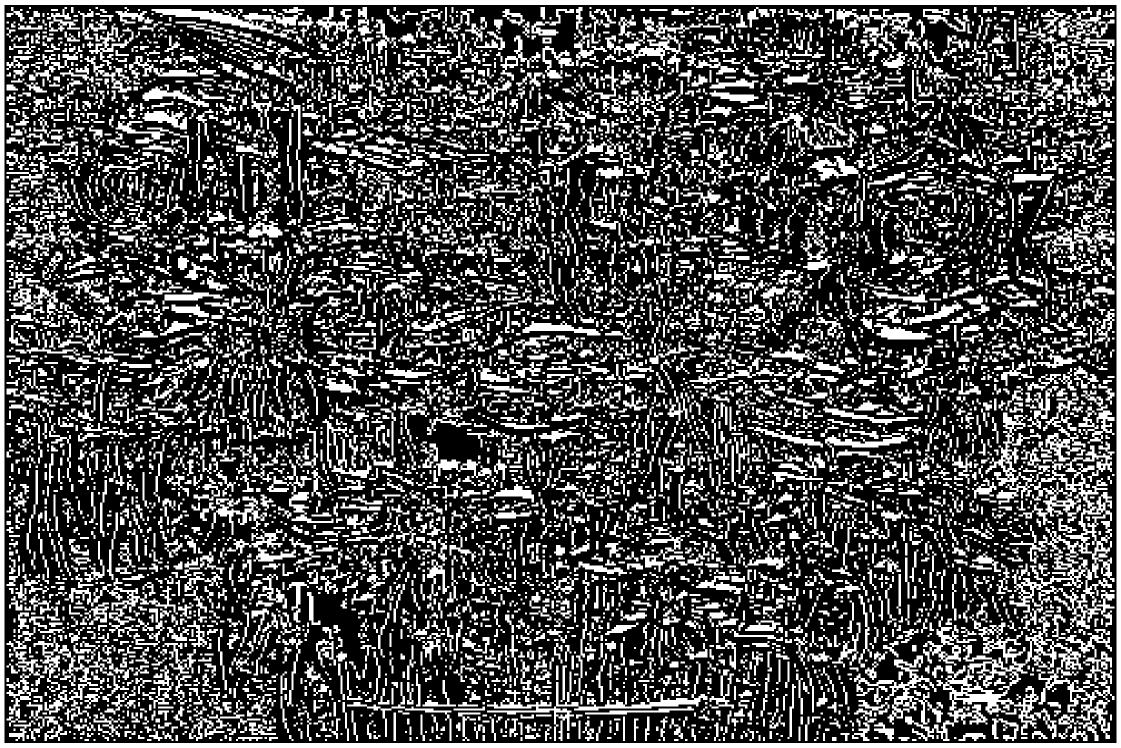
Figure 8. I_n of Flowers.jpg with Gaussian Sigma = 1

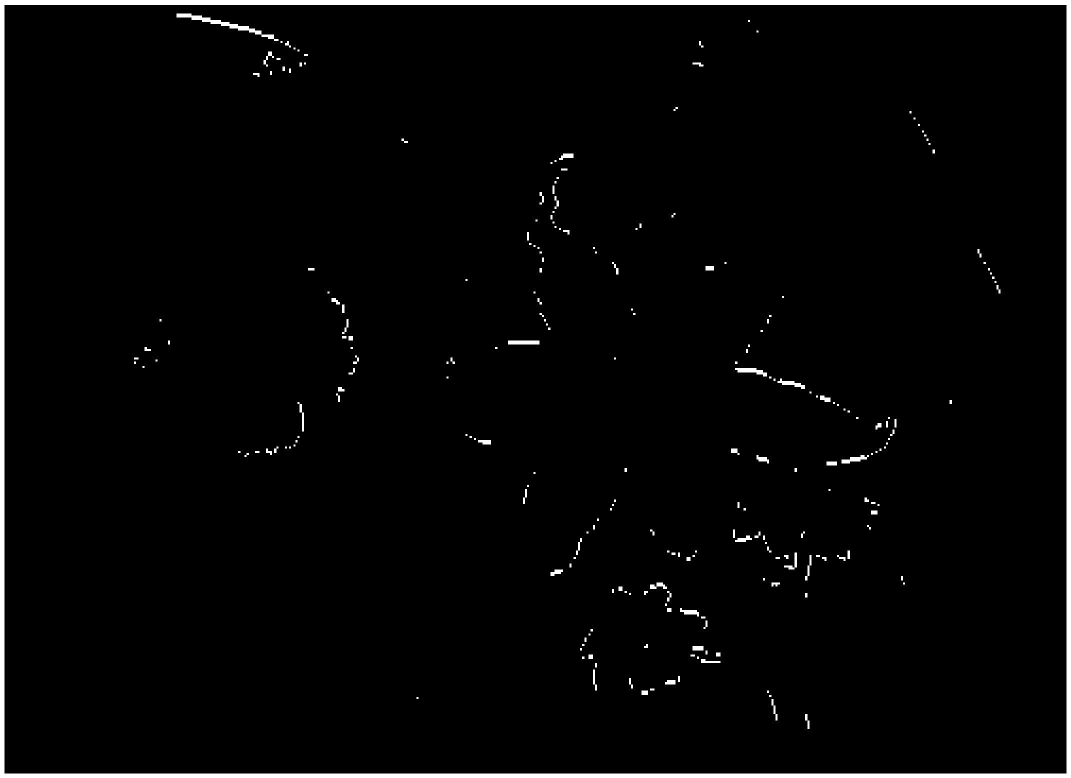

Figure 9. Out1HT(from hysteresis threshold) with Th = 100, Tl = 1

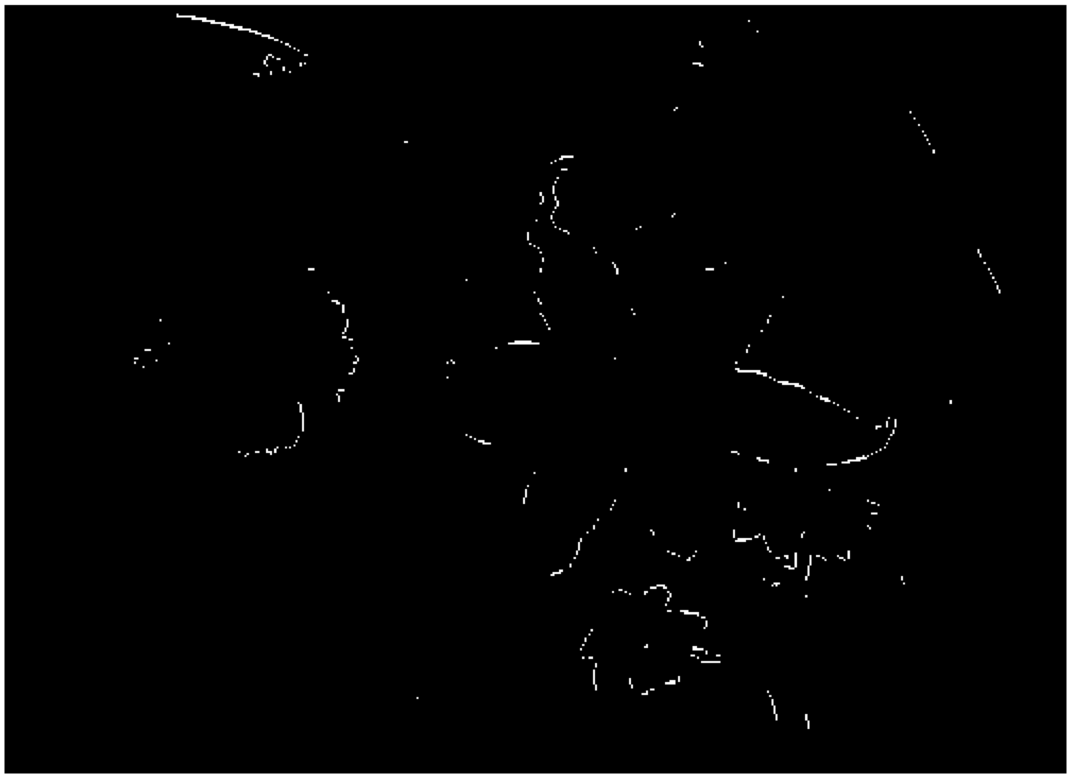

Figure 10. Out1HT(from hysteresis threshold) with Th = 100, Tl = 90

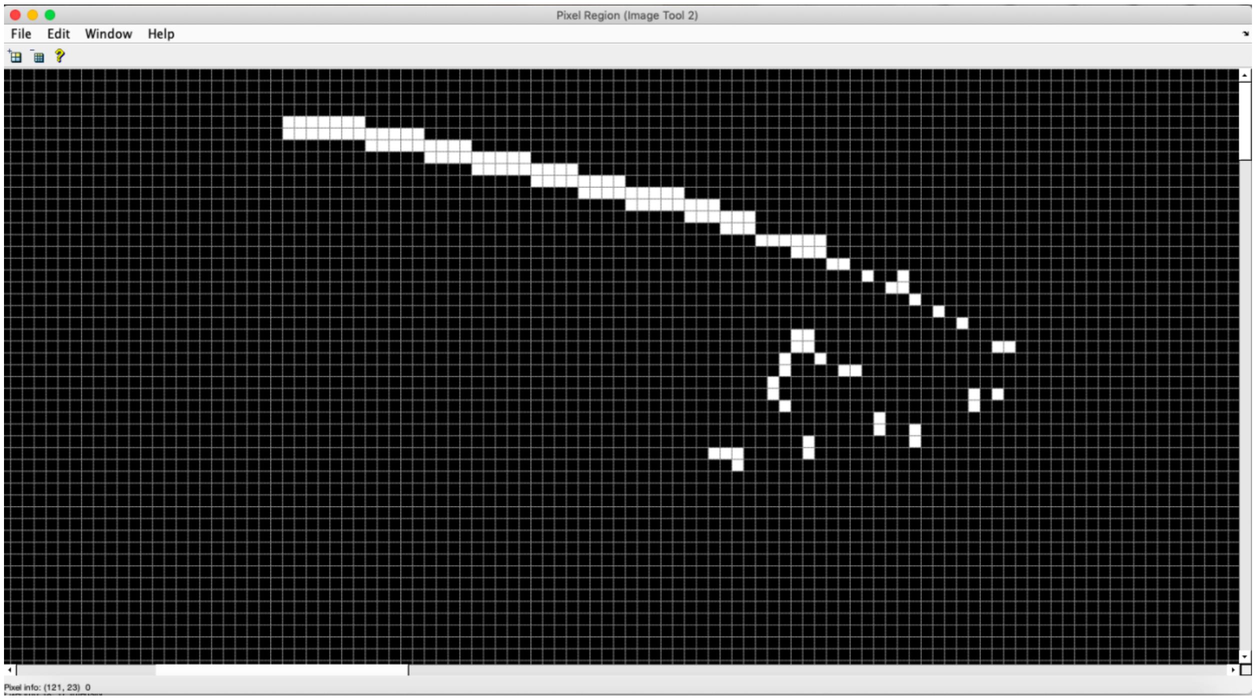

Figure 11. Pixel region of Out1HT(from hysteresis threshold) with Th = 100, Tl = 1

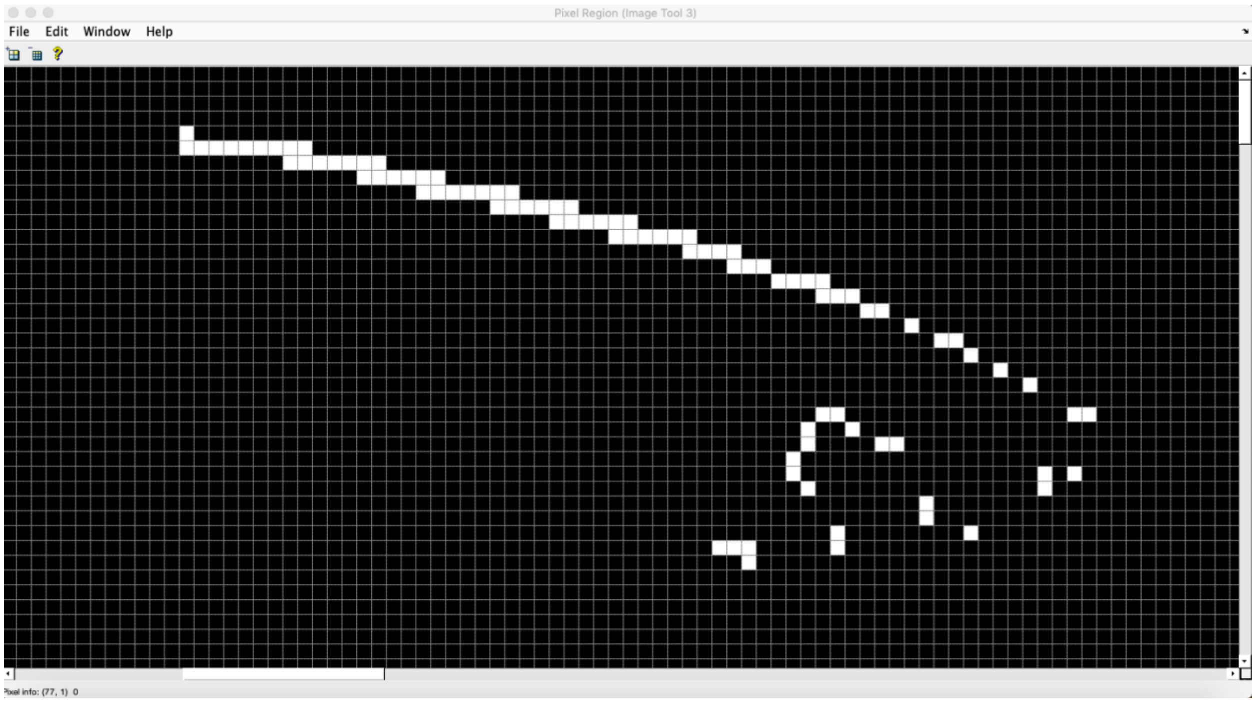

Figure 12. Pixel region of Out1HT(from hysteresis threshold) with Th = 100, Tl = 90

As continuation from last set of images where I compared the different set of Tl values, I tried an more extreme values for Flowers.jpg. Again, by simply looking at the figures 9 and 10, it is really difficult to notice the effects of different Tl values. However, if we compare their pixel regions, we can see that with higher Tl value, the edge is thinner than the one with low Tl value.

Figure 13 shows a final output for Flowers.jpg to conclude this set of outputs.

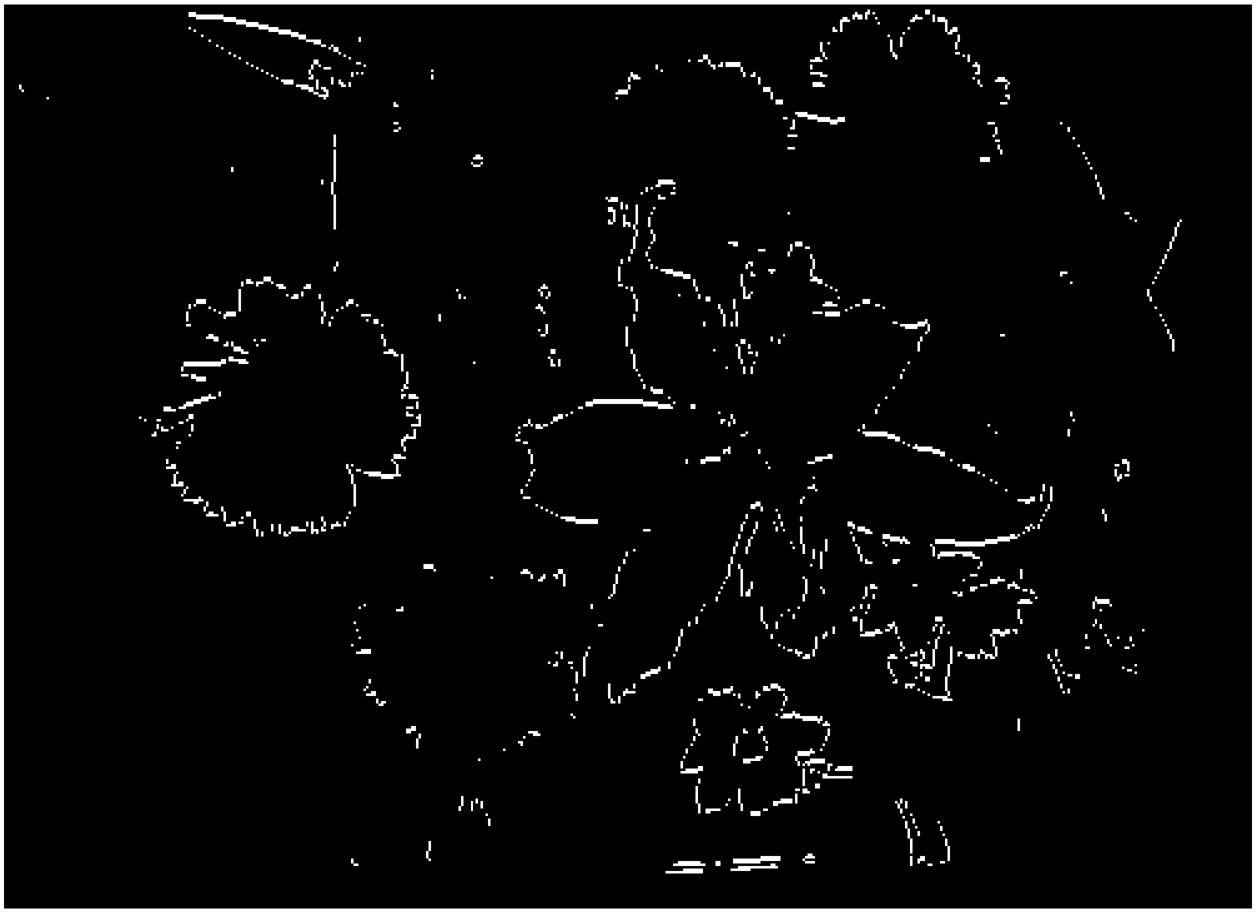

Figure 13. Out1HT(from hysteresis threshold) with Th = 60, Tl = 20

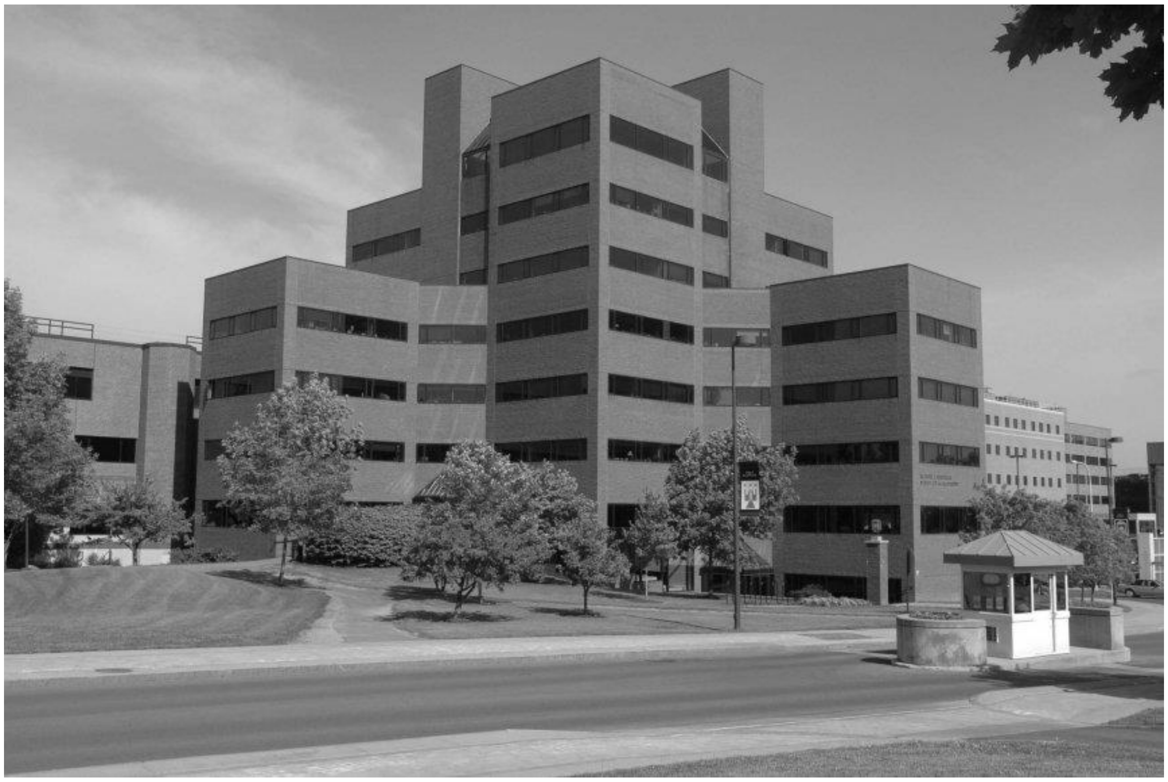

Figure 14. Original image of ‘Syracuse_02.jpg’

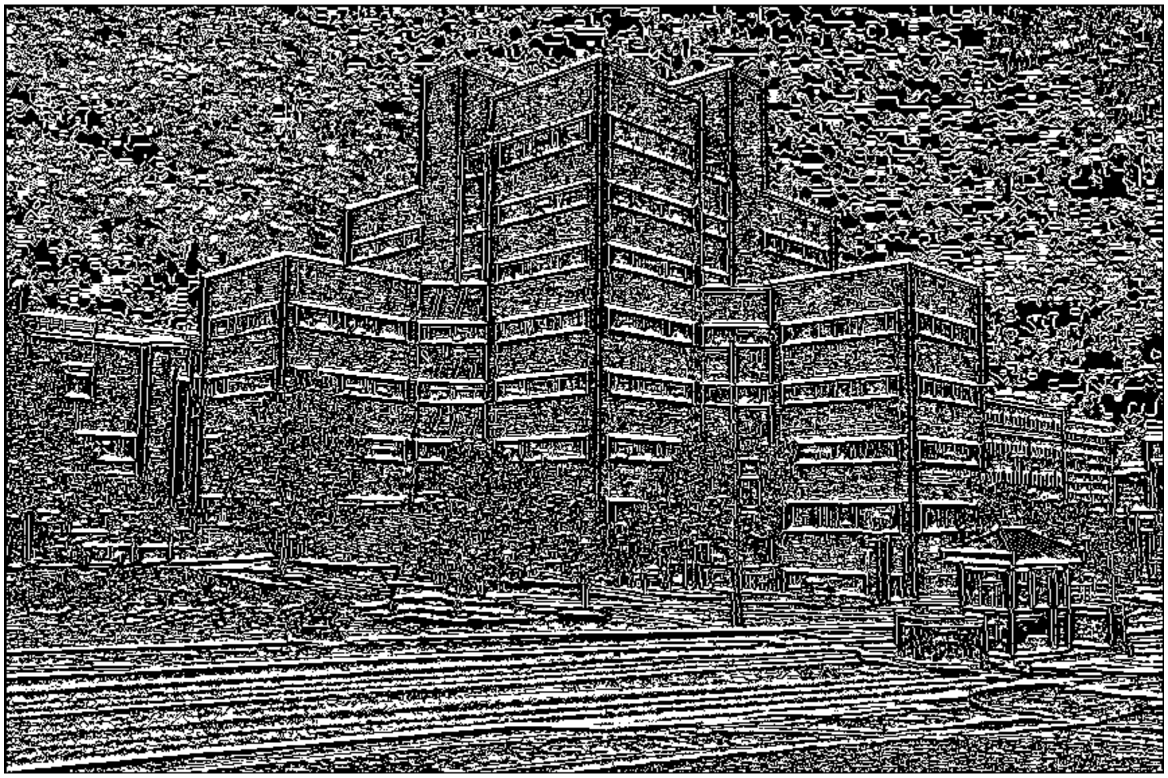

Figure 15. I_n of Syracuse_02.jpg with Gaussian Sigma = 1

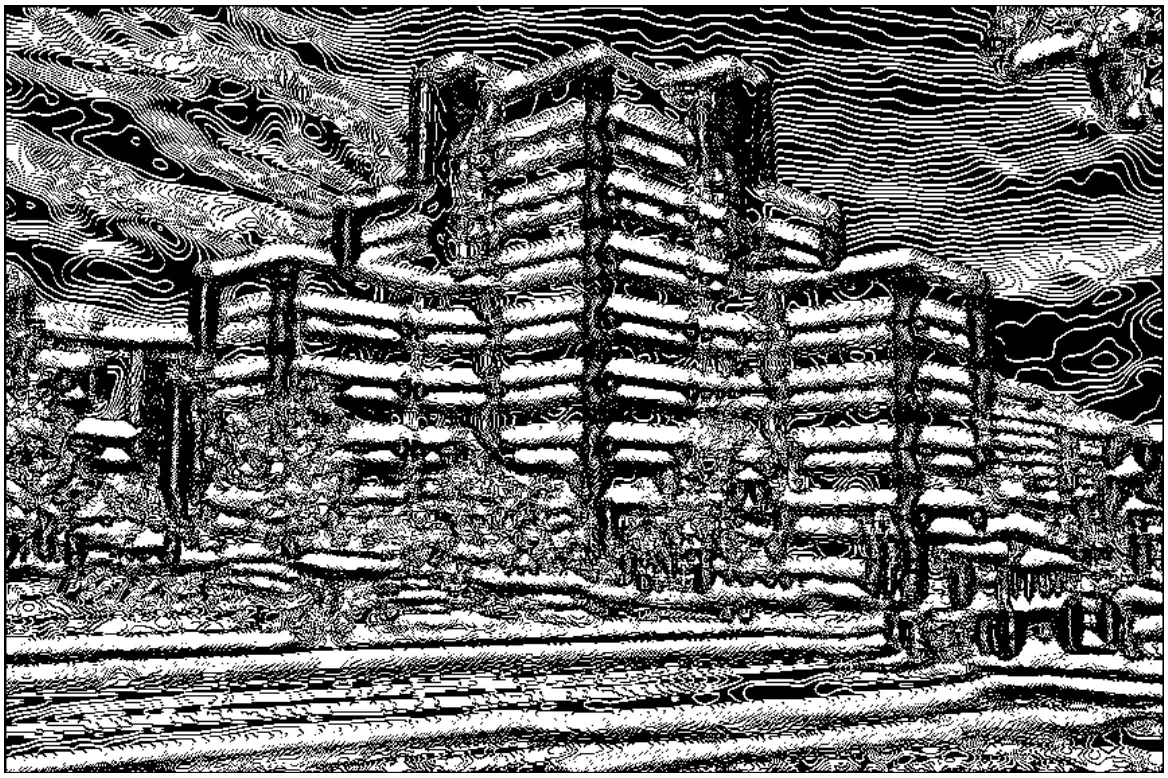

Figure 16. I_n of Syracuse_02.jpg with Gaussian Sigma = 5

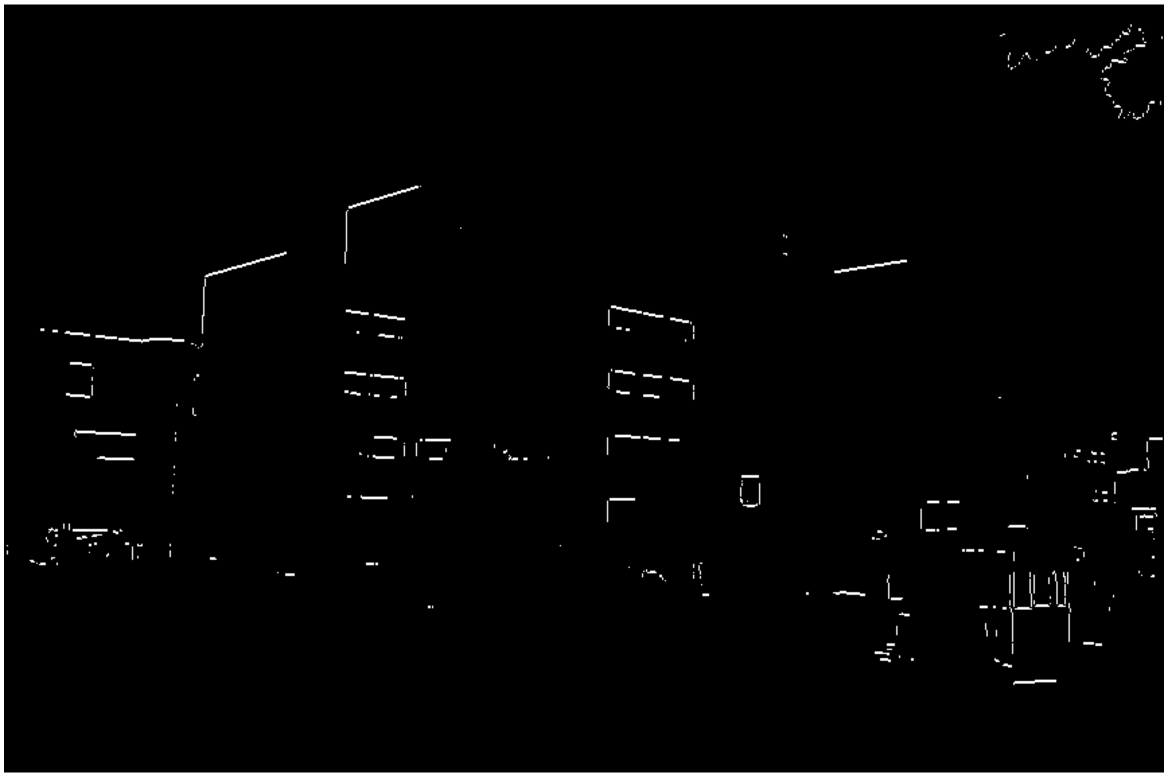

Figure 17. Out3HT(from hysteresis threshold) with Th = 60, Tl = 20

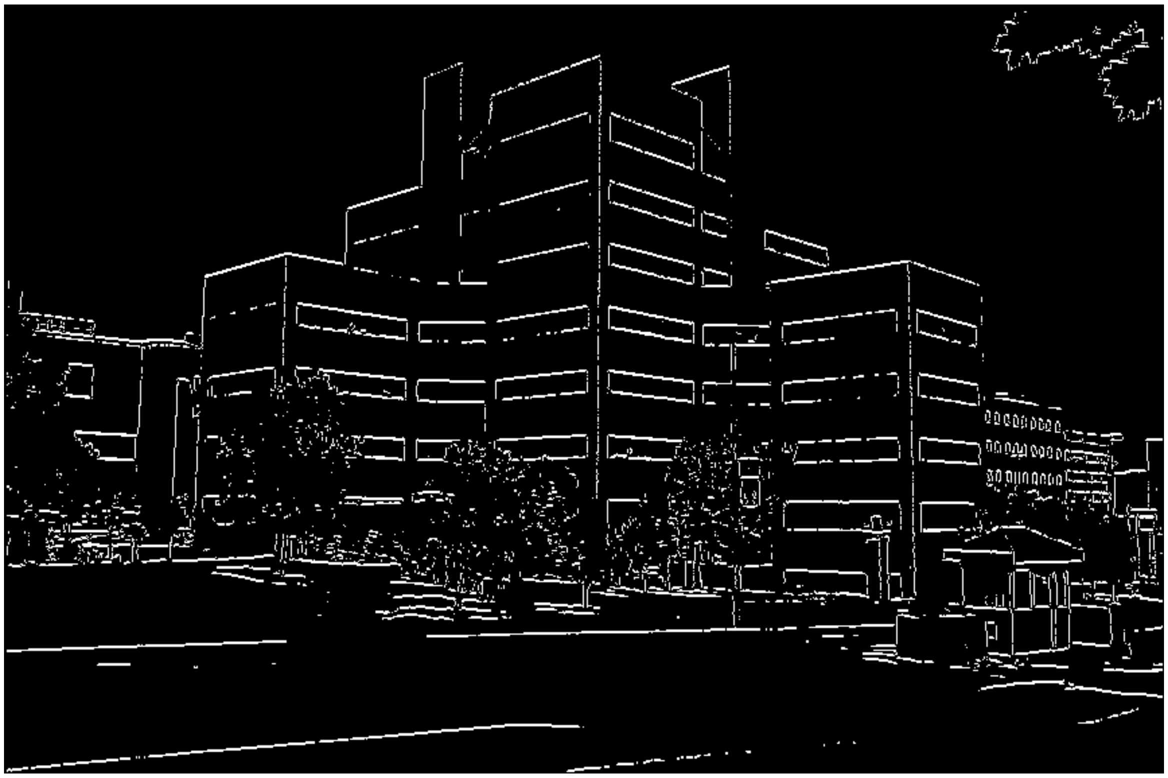

Figure 18. Out3HT(from hysteresis threshold) with Th = 30, Tl = 20

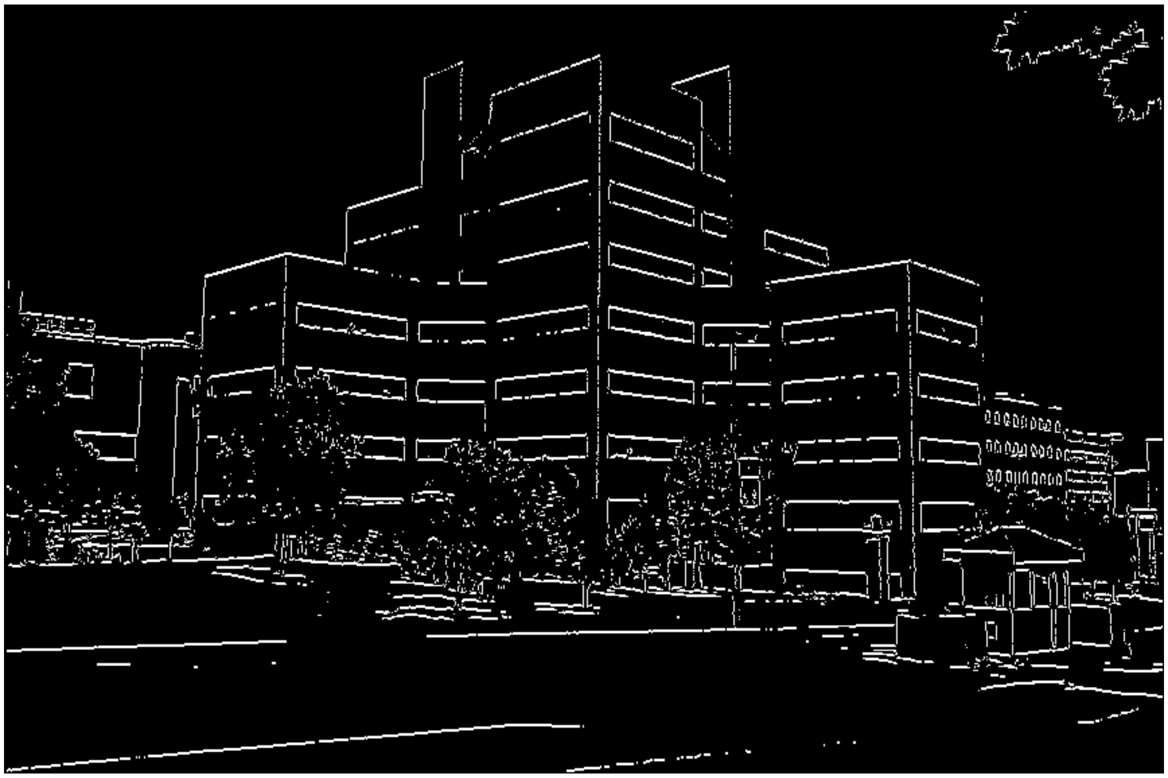

Figure 19. Out3HT(from hysteresis threshold) with Th = 30, Tl = 1

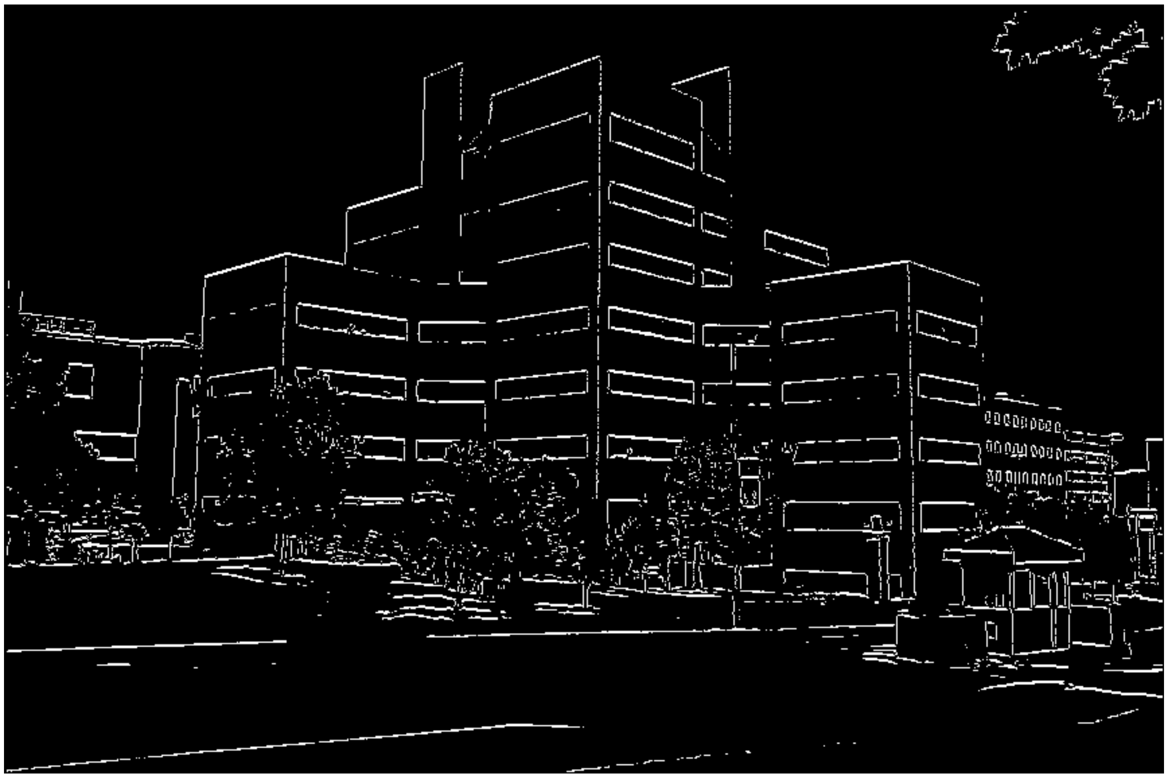

Figure 20. Out3HT(from hysteresis threshold) with Th = 30, Tl = 29

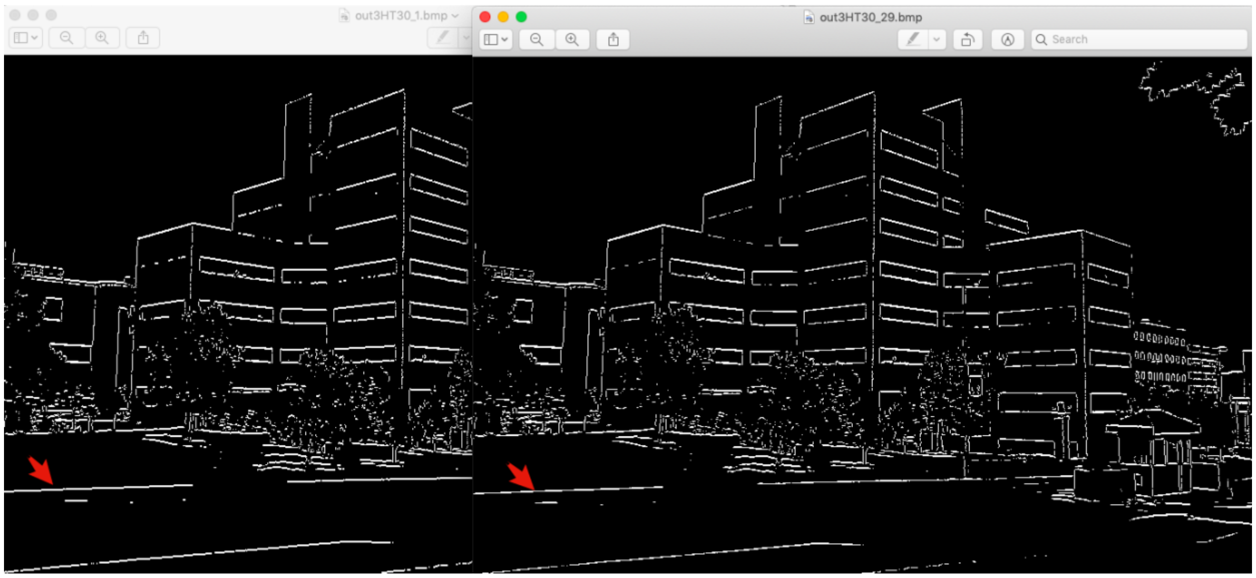

Figure 21. Comparison between figures 19 and 20

The two images of non-max suppression from figures 15 and 16 supports my observation from figures 2 and 3, where larger sigma value blurs the image and makes the edges too wide to be processed.

Figure 17 and 18 also supports that if the Th is too high, then there is less strong points to construct the edges, so the number of visible edges is less.

Figure 19 and 20 compares the effect of Tl values. The larger the Tl value, the thinner the edge.

Figure 21 shows a side by side comparison of figures 19 and 20.

An Image of My Own

I specifically picked an image where there are lots of edges.

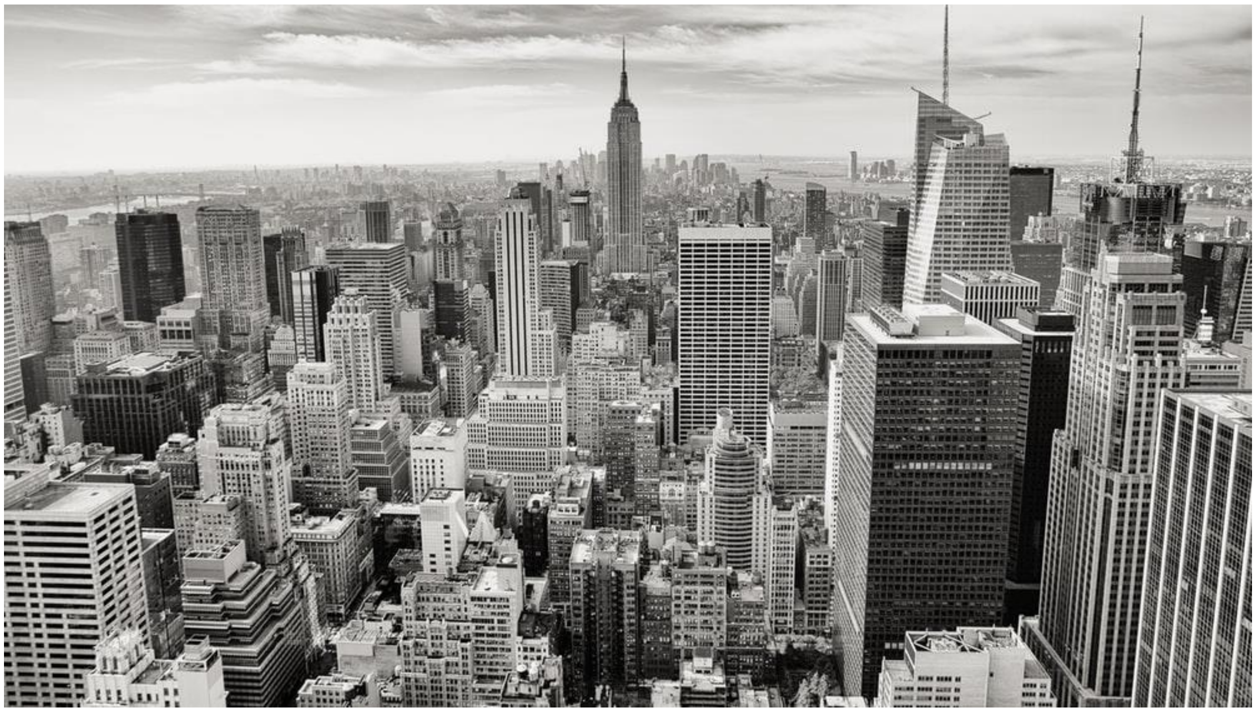

Figure 22. Original image of ‘NYC.jpg’

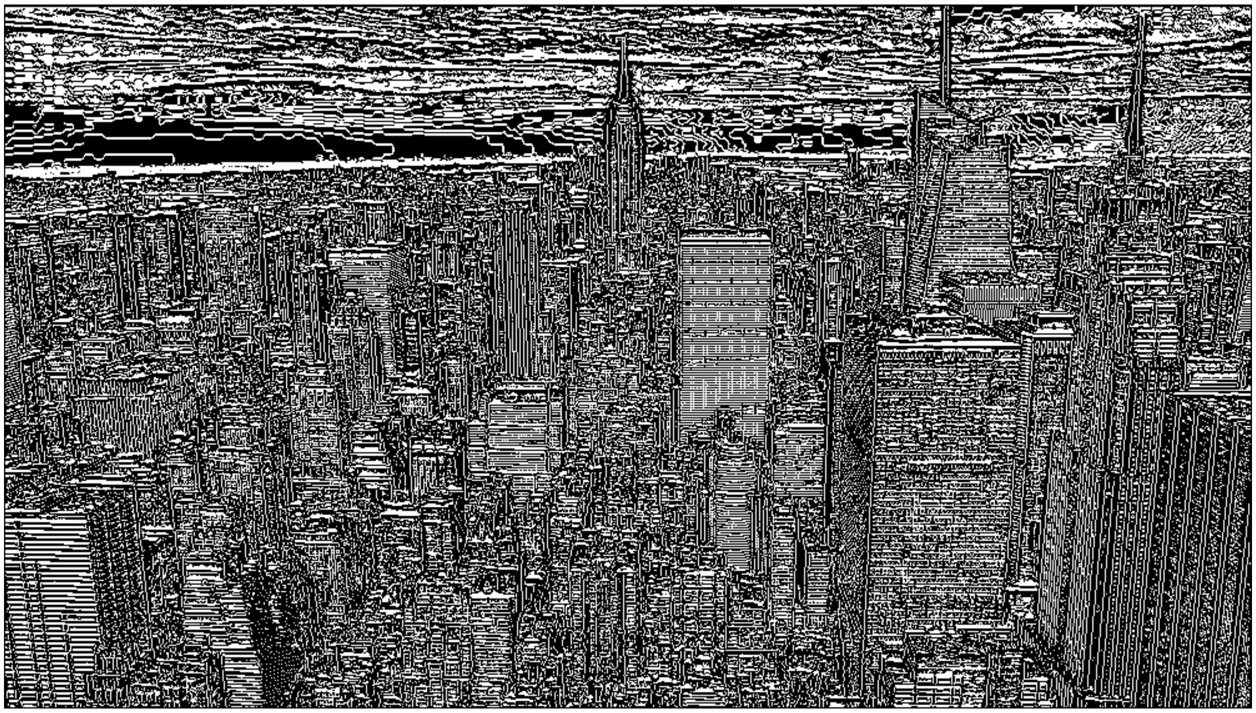

Figure 23. I_n of NYC.jpg with Gaussian Sigma = 1

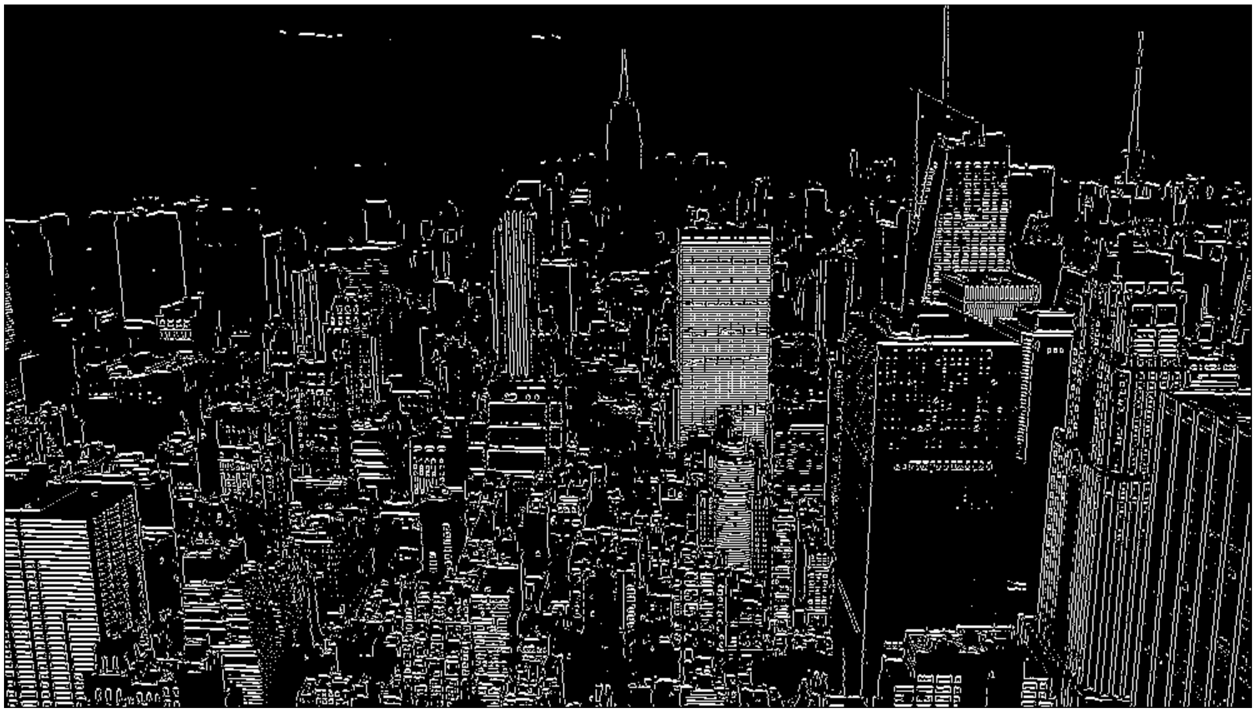

Figure 24. OutNYCHT(from hysteresis threshold) with Th = 40, Tl = 20

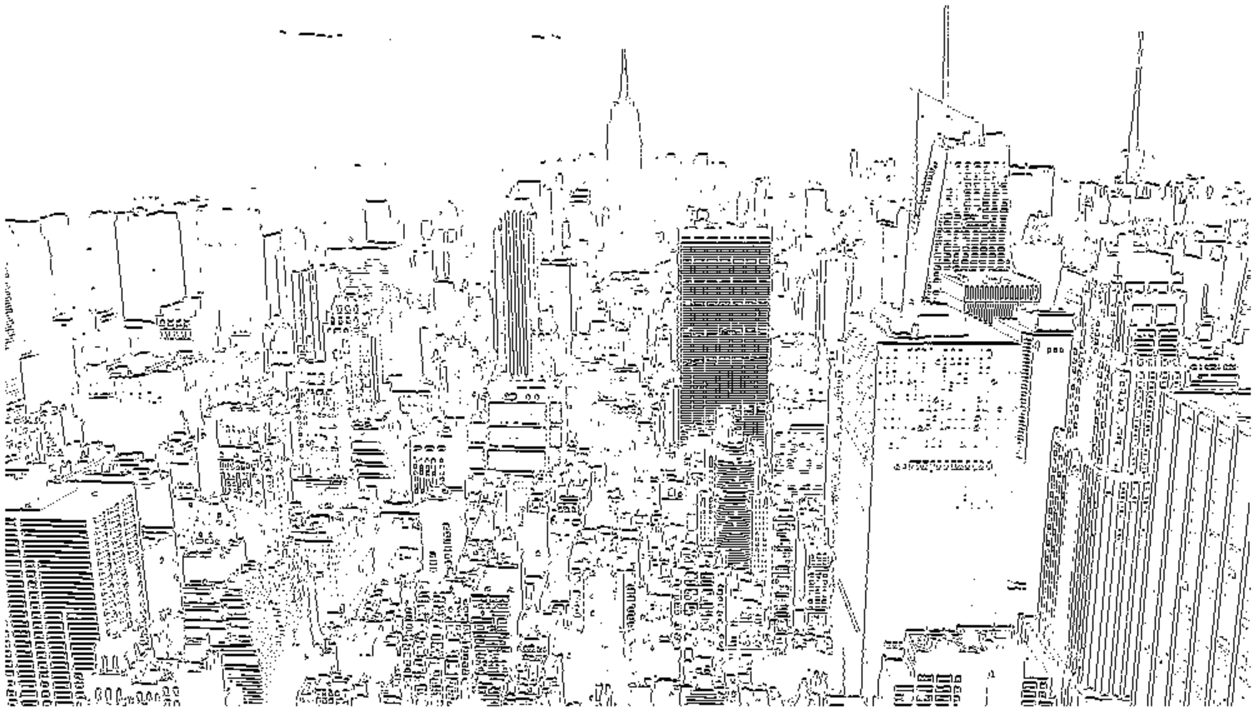

Figure 25. The reverse of figure 24

Future Improvement

Since the output highly depends on the gaussian filter sigma and high and low threshold values, it is hard to say if an output is right or wrong. One possible improvement it to find the optimum values to get the best result possible. The grayscale differs from image to image, then maybe finding the range of average grayscale values might help us to determine which threshold and sigma values to choose.

Conclusion

The gaussian filter sigma affects all further processes of the image, so it must be chosen carefully at the beginning. T_h value determines how many points are strong points and can be used to build the edge. T_l value decides how thin the edge is and how many points are not considered as edge.