CSC420 Assignmnet 1

Xiaoyu Zhou, 1004081147

October 15, 2020

Q1: LTI Systems and Convolution

$x(n) = \int_{-\infty}^{\infty} \delta(n-i)x(i)$
$= \sum_{i=-\infty}^{\infty} \delta(n-i)x(i)$
So, $T(x(n)) = T(\sum_{i=-\infty}^{\infty} \delta(n-i)x(i))$
$= \sum_{i=-\infty}^{\infty} T(\delta(n-i)x(i))$
$= \sum_{i=-\infty}^{\infty} x(i)T(\delta(n-i))$ (By the property of linear system)
$= \sum_{i=-\infty}^{\infty} x(i)h(n-i)$ (By the property of time-invariant)
$= h(n) * x(n)$

Q2: Polynomial Multiplication and Convolution

Let $m = length(\mathbf{u})$ and $n = length(\mathbf{v})$.
Let $\mathbf{w}$ be the result of convolution of $\mathbf{u}$ and $\mathbf{v}$, where $\mathbf{w}$ is a vector with length $m+n-1$.
By the calculation of convolution, we can know the k^{th} element of $\mathbf{w}$ is: $\mathbf{w}(k) = \sum_j \mathbf{u}(j)\mathbf{v}(k-j+1)$
Then, let r be the result of the multiplication of the polynomials represented by $\mathbf{u}$ and $\mathbf{v}$.
$r = \mathbf{u}(0) * \mathbf{v}(0) * x^{m+n} + [(\mathbf{u}(0) * \mathbf{v}(1) + \mathbf{u}(1) * \mathbf{v}(0)) * x^{m+n-1}] + ...$
The length of $\mathbf{r}$ is also $m+n-1$. Also, the coefficients of the k^{th} element in $\mathbf{r}$ is $\sum_j \mathbf{u}(j)\mathbf{v}(k-j+1)$.
Thus, we can see that convolving $\mathbf{u}$ and $\mathbf{v}$ is equivalent to multiplying the two polynomials they each represent.

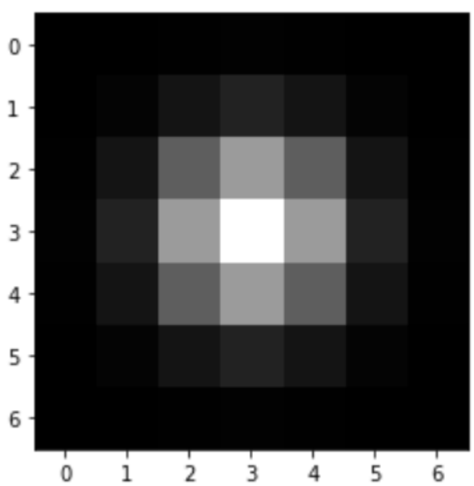
Q3: Laplacian Operator

Let the original point be (x, y) and the point after rotating θ degrees be (u, v).
We can know that $u = xcos\theta + ysin\theta$ and $v = -xsin\theta + ycos\theta$
To prove Laplacian is rotation invariant, we want to show that $\frac{d^2f}{dx^2} + \frac{d^2f}{dy^2} = \frac{d^2f}{du^2} + \frac{d^2f}{dv^2}$.
We can easily know :$\frac{du}{dx} = cos\theta$, $\frac{dv}{dx} = -sin\theta$, $\frac{du}{dy} = sin\theta$, $\frac{du}{dy} = cos\theta$
So, $\frac{df}{dx} = \frac{df}{du}\frac{du}{dx} + \frac{df}{dv}\frac{dv}{dx} = \frac{df}{du}cos\theta - \frac{df}{dv}sin\theta$
$\frac{d^2f}{dx^2}$
$= \frac{d}{dx}\frac{df}{dx}$
$= \frac{d}{dx}(\frac{df}{du}cos\theta - \frac{df}{dv}sin\theta)$
$= \frac{d}{dx}\frac{df}{du}cos\theta - \frac{d}{dx}\frac{df}{dv}sin\theta$
$= \frac{d}{du}\frac{df}{dx}cos\theta - \frac{d}{dv}\frac{df}{dx}sin\theta$

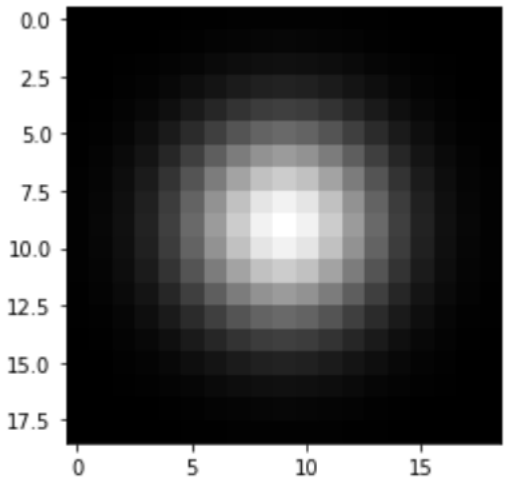
$= \frac{d}{du}(\frac{df}{du}cos\theta - \frac{df}{dv}sin\theta)cos\theta - \frac{d}{dv}(\frac{df}{du}cos\theta - \frac{df}{dv}sin\theta)sin\theta$
$= \frac{d^2f}{du^2}cos^2\theta - \frac{d}{du}\frac{df}{dv}cos\theta sin\theta - \frac{d}{dv}\frac{df}{du}cos\theta sin\theta + \frac{d^2f}{dv^2}sin^2\theta$
$= \frac{d^2f}{du^2}cos^2\theta - 2\frac{d}{du}\frac{df}{dv}cos\theta sin\theta + \frac{d^2f}{dv^2}sin^2\theta$
Same for y: $\frac{df}{dy} = \frac{df}{du}\frac{du}{dy} + \frac{df}{dv}\frac{dv}{dy} = \frac{df}{du}sin\theta + \frac{df}{dv}cos\theta$
$\frac{d^2f}{dy^2}$
$= \frac{d}{dy}\frac{df}{dy}$
$= \frac{d}{dy}(\frac{df}{du}sin\theta + \frac{df}{dv}cos\theta)$
$= \frac{d}{dy}\frac{df}{du}sin\theta + \frac{d}{dy}\frac{df}{dv}cos\theta$
$= \frac{d}{du}\frac{df}{dy}sin\theta + \frac{d}{dv}\frac{df}{dy}cos\theta$
$= \frac{d}{du}(\frac{df}{du}sin\theta + \frac{df}{dv}cos\theta)sin\theta + \frac{d}{dv}(\frac{df}{du}sin\theta + \frac{df}{dv}cos\theta)cos\theta$
$= \frac{d^2f}{du^2}sin^2\theta + \frac{d}{du}\frac{df}{dv}cos\theta sin\theta + \frac{d}{dv}\frac{df}{du}cos\theta sin\theta + \frac{d^2f}{dv^2}cos^2\theta$
$= \frac{d^2f}{du^2}sin^2\theta + 2\frac{d}{du}\frac{df}{dv}cos\theta sin\theta + \frac{d^2f}{dv^2}cos^2\theta$
Thus, we can get :
$\frac{d^2f}{dx^2} + \frac{d^2f}{dy^2}$
$= (\frac{d^2f}{du^2}cos^2\theta - 2\frac{d}{du}\frac{df}{dv}cos\theta sin\theta + \frac{d^2f}{dv^2}sin^2\theta) + (\frac{d^2f}{du^2}sin^2\theta + 2\frac{d}{du}\frac{df}{dv}cos\theta sin\theta + \frac{d^2f}{dv^2}cos^2\theta)$
$= \frac{d^2f}{du^2}(sin^2\theta + cos^2\theta) + \frac{d^2f}{dv^2}(sin^2\theta + cos^2\theta)$
$= \frac{d^2f}{du^2} + \frac{d^2f}{dv^2}$
Thus, we can see that Laplacian is rotation invariant.

Q4: Edge Detection

Please see the code in q4.py
Step1 Gaussian Blurring
For $\sigma = 1$ and filter size = 7:

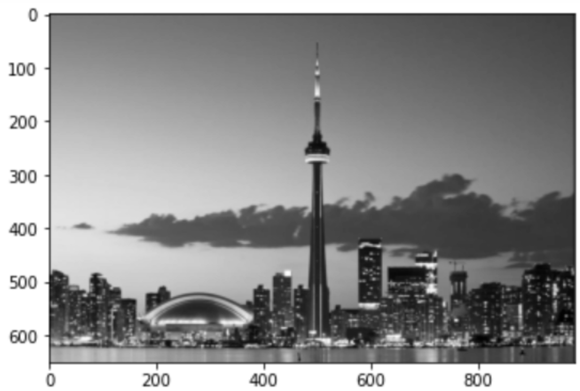


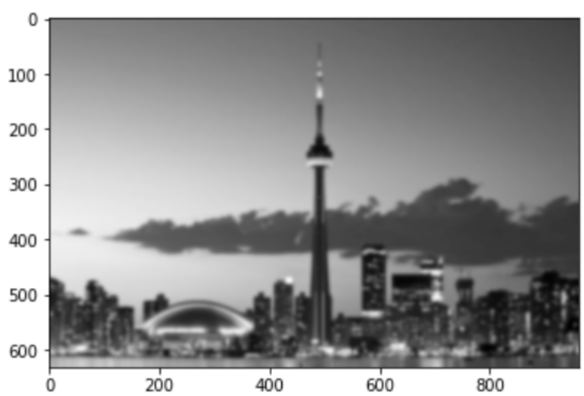
For $\sigma = 3$ and filter size = 19:

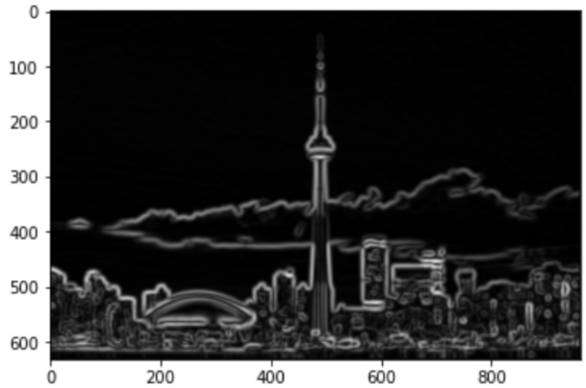


Step4 Test

Image 1 (in the order from original image to step1-3:







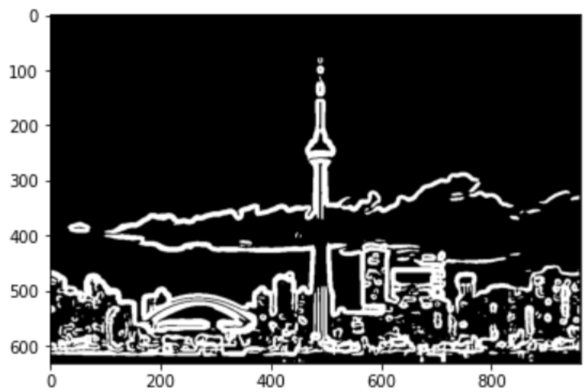
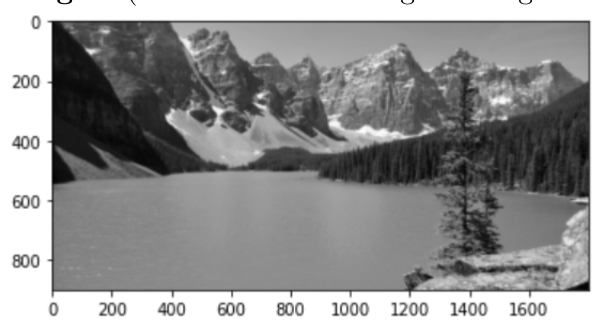
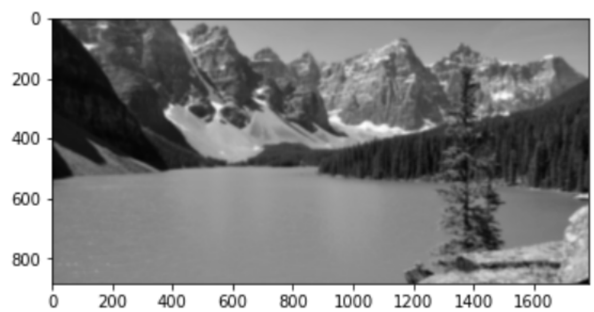
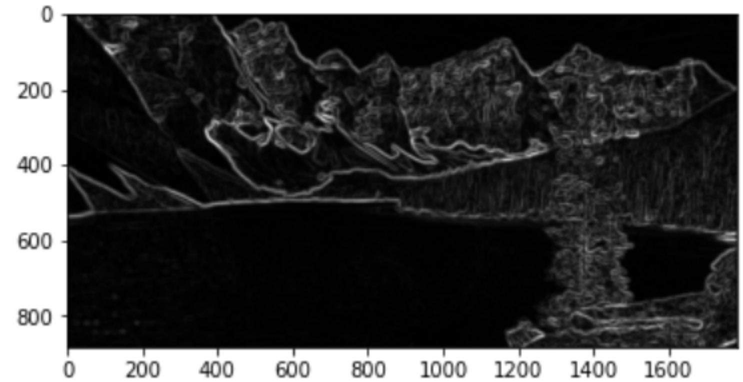


Image 2 (in the order from original image to step1-3:







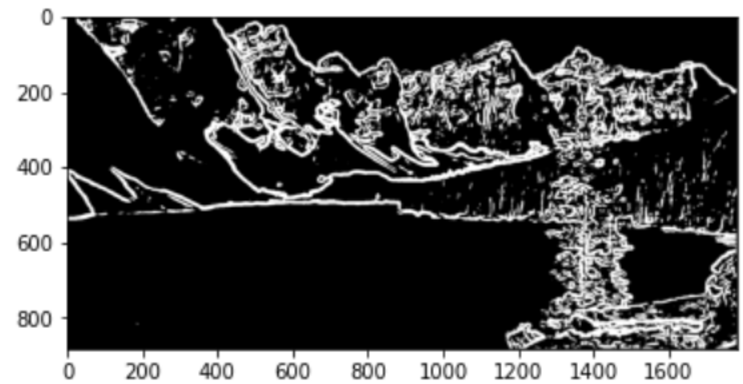
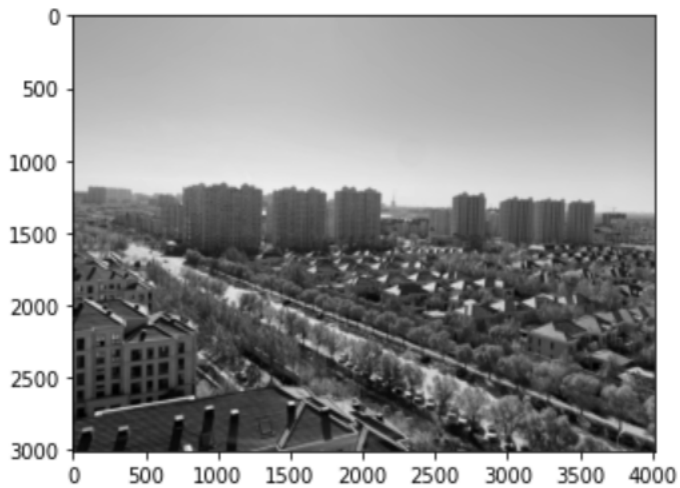
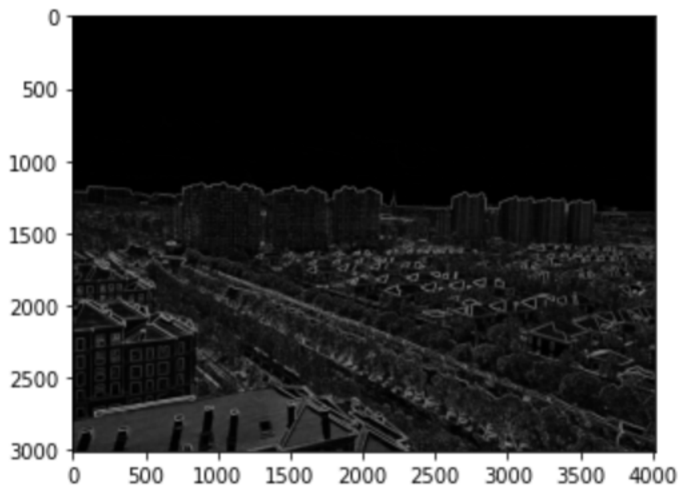
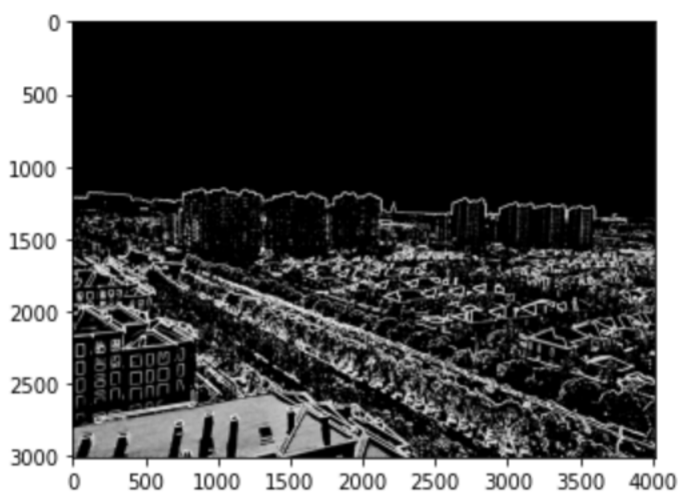


Image of my own (in the order from original image to step1-3:







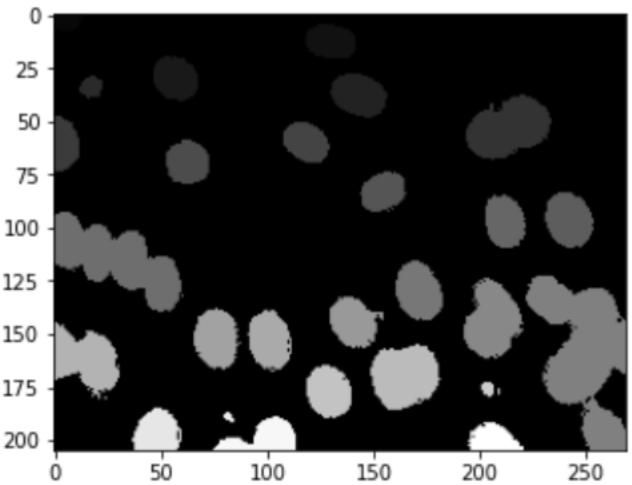
how algorithm works: First of all, the Gaussian blurring make the picture indistinct. Then, find the gradient of the image. Finally, through gradient, we can find the edge image of the original image. The advantage of the algorithm is it is easy to compute do not have too many steps. However, the algorithm is sensitive to the noise.

Q5:Connected-component Labeling

Please see the code in q5.py

Q6: Count number of cells

Here is the image that I got after iterate the algorithm in Q5:



And the number of cells that the algorithm estimate is 30.

My result is closed to the real number. I suppose that the reason that my estimation is less than the real number is my algorithm cannot ditinguish the black pixedls well. Since in the picture, all most none of black pixel are 0, so I set the black pixel to be less than 10. So, may be I can make a more accurate division for the for the black and white pixels.