CSC 420 A1

Yinjun Zheng

Sep, 2021

1 Part 1

1.1 Q1

We know that $x(n) = \sum_{k=-\infty}^{\infty} x(k)\delta(n-k)$, so

$$\begin{aligned} T[x(n)] &= T[\sum_{k=-\infty}^{\infty} x(k)\delta(n-k)] \\ &= \sum_{k=-\infty}^{\infty} T[x(k)\delta(n-k)] \\ &= \sum_{k=-\infty}^{\infty} x(k)T[\delta(n-k)] \\ &= \sum_{k=-\infty}^{\infty} x(k)h(n-k) \\ &= h(n) * x(n) \end{aligned}$$

Therefore, $T[x(n)] = h(n) * x(n)$

1.2 Q2

Let $u \in R^{m+1}, v \in R^{n+1}$, assume $m > n$, that is:
$p_1 = u_m x^m + u_{m-1}x^{m-1} + ... + u_1 x + u_0$
$p_2 = v_n x^n + v_{n-1}x^{n-1} + ... + v_1 x + v_0$
Add padding to u,v to make them into R^{m+n+1}, that is:
$u = (0, 0, ..., 0, u_m, u_{m-1}, ..., u_1, u_0) \in R^{m+n+1}$
$v = (0, 0, ..., 0, v_n, v_{n-1}, ..., v_1, v_0) \in R^{m+n+1}$

Let $p_1 p_2 = \sum_{i=0}^{m+n} c_i x^i$, then

$$\begin{aligned} c_i x^i &= u_0 x^0 * v_i x^i + u_1 x^1 * v_{i-1} x^{i-1} + ... + u_i x^i * v_0 x^0 \\ &= \sum_{j=0}^{i} u_j v_{i-j} x^i \\ &= (u * v)_i x^i \end{aligned}$$

Therefore, $c = u * v$
The coefficients of the multiplication is the convolution of u and v.

1.3 Q3

Claim: The minimum information we need to know from the Gaussian Pyramid is the value of I_n
Suppose in each step i, we blur image I_k with operation B_i, and downsample the blurred image by factor two with operation D_i, and define the inverse of D_i as upsample U_{i+1}.
The we get the formula on I as follows:

$$I_{k+1} = D_k B_K I_k$$

Also by the definition of Laplacian pyramid, we have:

$$I_k - B_K I_k = L_k$$

Therefore, we get the recursion formula as follows:

$$I_k - U_{k+1} I_{k+1} = L_k$$

$$I_k = U_{k+1} I_{k+1} + L_k$$

Therefore, as long as we have the value of $\mathrm{I}_n, wecangetthe\mathrm{I}_0$ by the recursion formula.
We get the closed formula for I_0 as follows:

$$\begin{aligned} I_0 &= U_1 I_1 + L_0 \\ &= U_1(U_2 I_2 + L_1) + L_0 \\ &= U_1 U_2 (U_3 I_3 + L_2) + U_1 L_1 + L_0 \\ &= ... \\ &= \prod_{i=0}^{n} U_i I_n + \sum_{j=0}^{n-1} (\prod_{k=0}^{j} U_j) L_j \end{aligned}$$

(2)

Therefore, the minimum information we need to know is I_n.
The closed expression for I_0 is: $I_0 = \prod_{i=0}^{n} U_i I_n + \sum_{j=0}^{n-1} (\prod_{k=0}^{j} U_j) L_j$, where U_i is the upsample operation at step i.

1.4 Q4

Suppose that we rotate (x,y) by θ to get (r, r'), then
$r = xcos\theta - ysin\theta, r' = xsin\theta + ycos\theta$
Need to show $\Delta I = I_{xx} + I_{yy} = I_{rr} + I_{r'r'}$
First, we have

$$\begin{aligned}
\frac{\partial I}{\partial x} &= \frac{\partial I}{\partial r}\frac{\partial r}{\partial x} + \frac{\partial I}{\partial r'}\frac{\partial r'}{\partial x} \\
&= \frac{\partial I}{\partial r}cos\theta + \frac{\partial I}{\partial r'}sin\theta \\
\frac{\partial I}{\partial y} &= \frac{\partial I}{\partial r}\frac{\partial r}{\partial y} + \frac{\partial I}{\partial r'}\frac{\partial r'}{\partial y} \\
&= -\frac{\partial I}{\partial r}sin\theta + \frac{\partial I}{\partial r'}cos\theta
\end{aligned}$$

Then we get the second derivative as follows:

$$\begin{aligned}
\frac{\partial^2 I}{\partial x^2} &= \frac{\partial}{\partial x}[\frac{\partial I}{\partial r}\frac{\partial r}{\partial x} + \frac{\partial I}{\partial r'}\frac{\partial r'}{\partial x}] \\
&= \frac{\partial}{\partial r}\frac{\partial I}{\partial x}cos\theta + \frac{\partial}{\partial r'}\frac{\partial I}{\partial x}sin\theta \\
&= \frac{\partial}{\partial r}(\frac{\partial I}{\partial r}cos\theta + \frac{\partial I}{\partial r'}sin\theta)cos\theta + \frac{\partial}{\partial r'}(\frac{\partial I}{\partial r}cos\theta + \frac{\partial I}{\partial r'}sin\theta)sin\theta \\
&= \frac{\partial I}{\partial r^2}cos^2\theta + \frac{\partial I}{\partial r\partial r'}2sin\theta cos\theta + \frac{\partial I}{\partial r'^2}sin^2\theta \\
\frac{\partial^2 I}{\partial y^2} &= \frac{\partial I}{\partial r^2}sin^2\theta - \frac{\partial I}{\partial r\partial r'}2sin\theta cos\theta + \frac{\partial I}{\partial r'^2}cos^2\theta
\end{aligned}$$

Therefore, we get

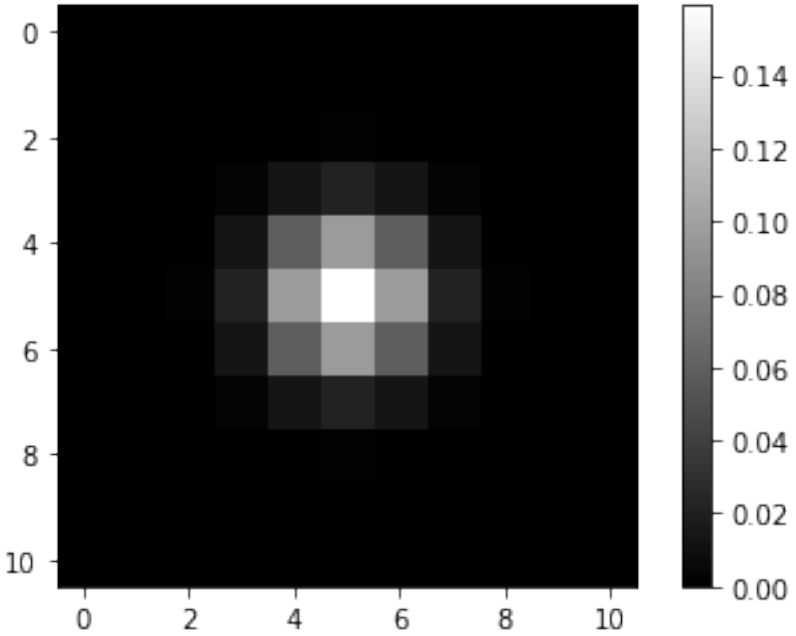
$$\begin{aligned}
\frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} &= \frac{\partial I}{\partial r^2}cos^2\theta + \frac{\partial I}{\partial r\partial r'}2sin\theta cos\theta + \frac{\partial I}{\partial r'^2}sin^2\theta + \frac{\partial I}{\partial r^2}sin^2\theta - \frac{\partial I}{\partial r\partial r'}2sin\theta cos\theta + \frac{\partial I}{\partial r'^2}cos^2\theta \\
&= \frac{\partial I}{\partial r^2}(cos^2\theta + sin^2\theta) + \frac{\partial I}{\partial r'^2}(sin^2\theta + cos^2\theta) \\
&= \frac{\partial^2 I}{\partial r^2} + \frac{\partial^2 I}{\partial r'^2}
\end{aligned}$$

Therefore, $\Delta I = I_{xx} + I_{yy} = I_{rr} + I_{r'r'}$
That is, Laplacian is rotation invariant.

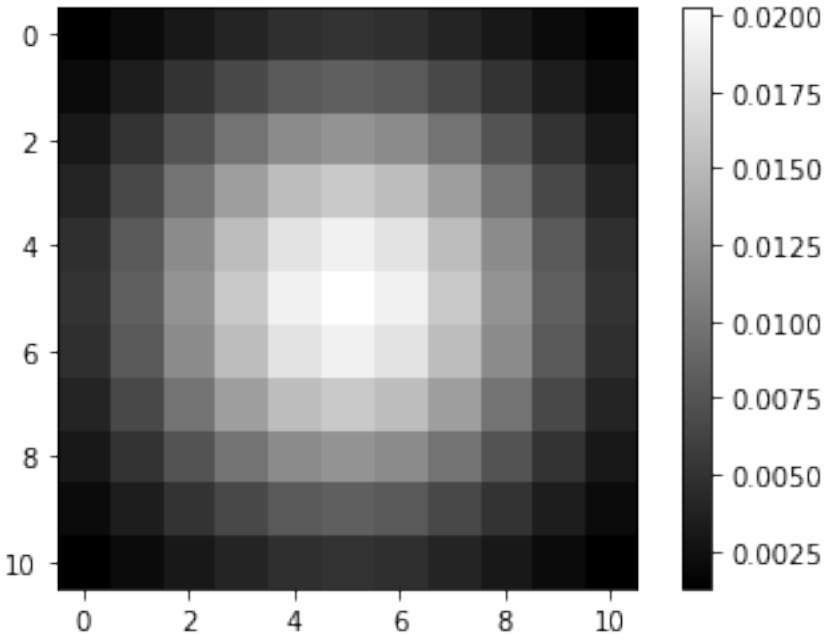
2 Part 2

2.1 Step 1

Please see the file Part2.ipynb for the code.
Visualization of 2D Gaussian Matrix are shown as follows:
(1) choose size=11 and $\delta = 1$



(2) choose size=11 and $\delta = 3$



2.2 Step 2

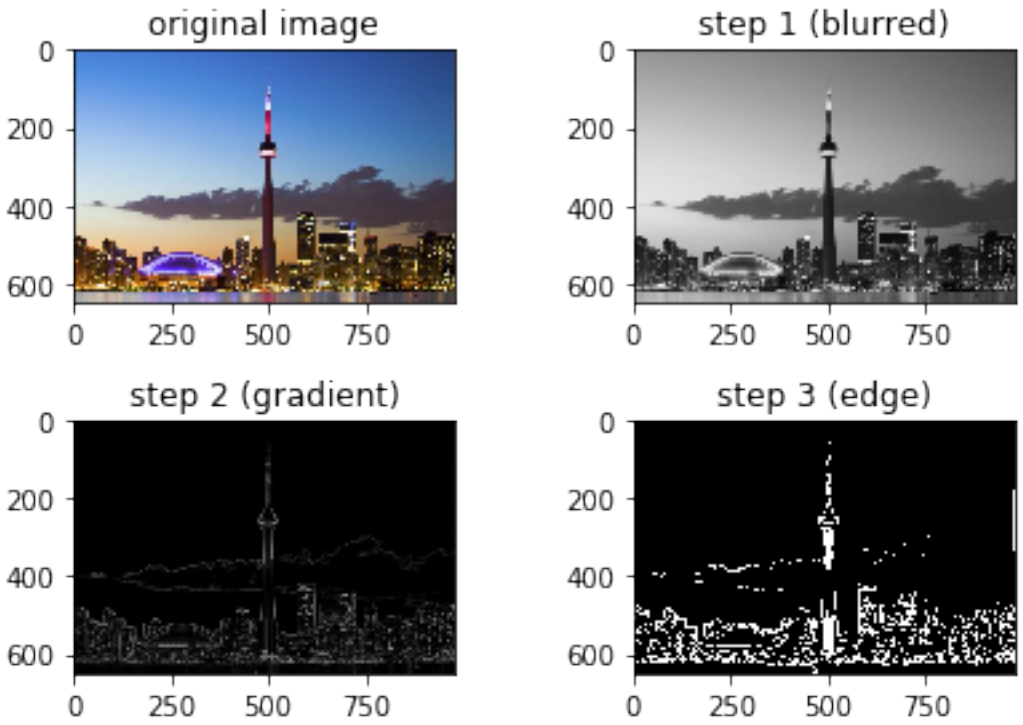
Please see the function convolution and function gradient_magnitude in the file Part2.ipynb for the code.

2.3 Step 3

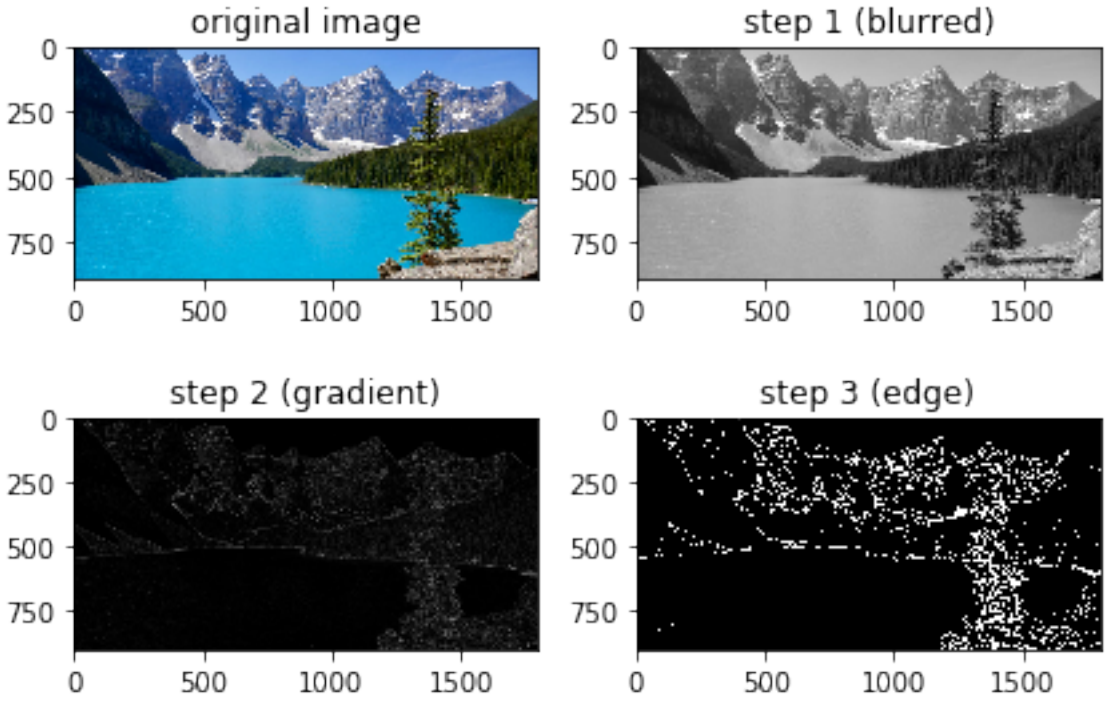
Please see the function set_threshold in the file Part2.ipynb for the code.

2.4 Step 4

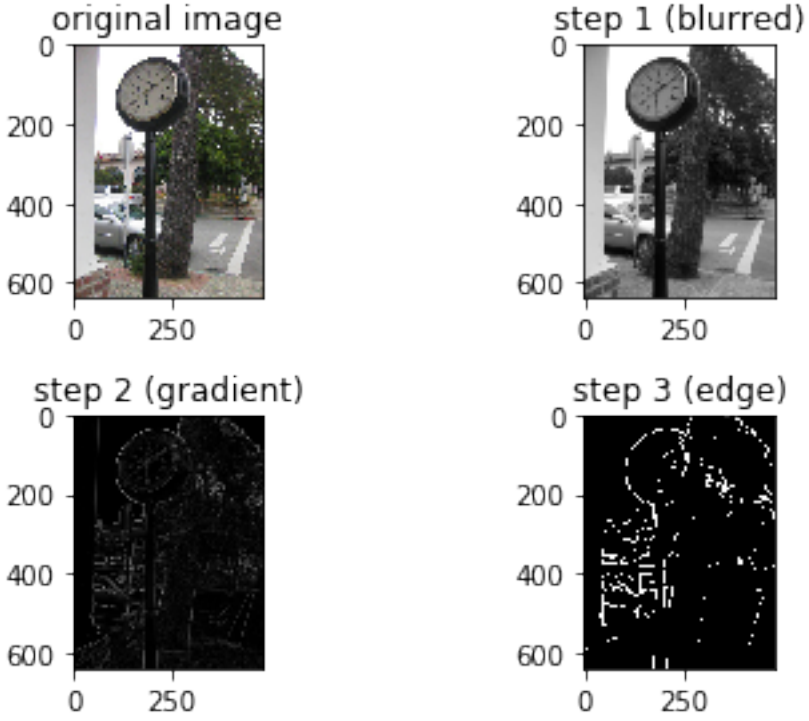
The visualization for image 1 is shown as follows:



The visualization for image 2 is shown as follows:



The visualization for the image I choose my self is shown as follows:



Discussion:
Strength: If we take a quick look at the result, the algorithm detects the edges of the original images quite well. It detects most of the clear edges. As we can see in images1, it detects the main edges of the tower and low towers. In image 2, it detects the edges of the lake and the mountains pretty good.

Weakness: The algorithm does not perform well on detecting edges where both sides have a similar color. For example, in image 1, it fails to distinguish between the tower and the cloud. Also it fails to show clear edges in the lower buildings. Similarly, in image 2, it fails to detect the edges between the mountains. It is probably because the image is blurred at the first step, we can make the edges clearer by increase delta, but it might no longer be smooth enough. Also the threshold is equal to find the median value of the gradient magnitude, we can improve it in some way. In a nutshell, the algorithm could be improved to detect some less obvious edges.