csc420 a1

Guanchen Zhang

September 2018

1 Question 1

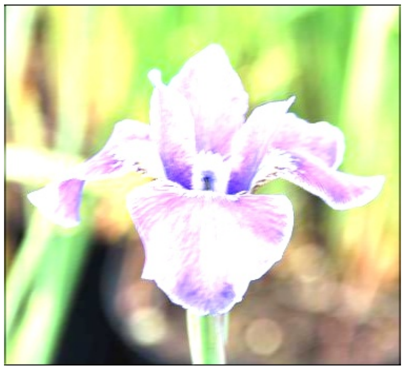


Figure 1: "full" mode

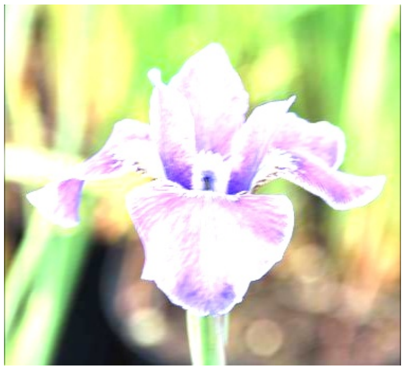


Figure 2: "same" mode

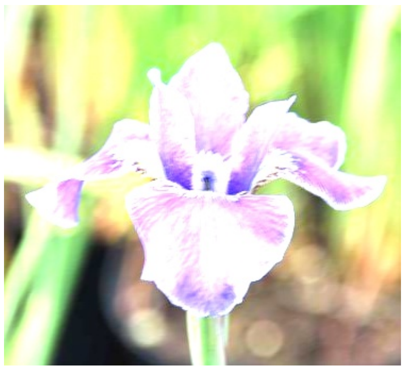


Figure 3: "Valid" mode

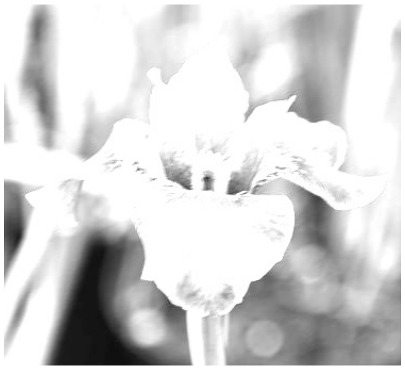


Figure 4: "Valid" mode with grayscale input image

The kernel we used in this question is $\begin{bmatrix} -0.25 & 0 & 0.25 \\ 1 & 0 & 1 \\ 0.25 & 0 & -0.25 \end{bmatrix}$

The input image has a dimension of 341*375*3. To do the correlation operation at the unknown pixels around the input image with "same" and "full" mode, I set these pixel to be 0, which is black in color. Therefore, the edges of the result images are slightly darker than neighbouring pixels. In this case, using the 3*3 kernel above, the result image generated with "full" model has a dimension of 343*377*3. The result image with "valid" mode has a dimension of 339*373*3. The result image with "same" mode has the same dimension as the input image. I also include an result image which took a grayscale image as input.

2 Question 2

Using the function in question 1 to calculate a convolution, we only need to flip the filter in x axis and y axis. The 2D gaussian filter was generated from the outer product of two 1D filter since 2D gaussian is separable. The result image is below:

3 Question 3

Convolution is a commutative operation because:

Question 2

Proof of commutative of Convolution

$f(n) * g(n) = \sum_{u=-K}^{K} f(u) \cdot g(n-u)$

Let $u' = n - u$

Then $u = n - u'$

since $u = [-K, K]$, then $u' = [n-k, n+k]$

From above, $\sum_{u=-k}^{k} f(n) \cdot g(n-u) = \sum_{u'=n-k}^{n+k} f(n-u') \cdot g(u')$

$= \sum_{u'=-k}^{k} g(u') f(n-u') = g(n) * f(n)$

$\therefore f(n) * g(n) = g(n) * f(n)$

Correlation is not a commutative operation because:

Let f' = flipping the filter f in both dimension

$f \otimes g = f' * g = g * f' = g' \otimes f'$

where g' is generated from flipping the filter g in both dimensions

Therefore, $f \otimes g \neq g \otimes f$

4 Question 4

The horizontal derivative of Gaussian filter G is a separable filter because it can be separated into horizontal and vertical components.

Guassian $f(x,y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{\sigma^2}\right)$

$$\frac{\partial f(x,y)}{\partial x} = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{\sigma^2}\right) \cdot \frac{\partial\left(-\frac{x^2+y^2}{\sigma^2}\right)}{\partial x}$$

$$= \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{\sigma^2}\right) \cdot \frac{(-2x)}{\sigma^2}$$

$$= \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{\sigma^2}} \cdot e^{\log_e -2x}$$

$$= \frac{1}{2\pi\sigma^2} e^{-\frac{y^2}{\sigma^2}} \cdot e^{-\frac{x^2}{\sigma^2} + \log_e -2x}$$

where the first term only depends on y and the second term only depends on x.

5 Question 5

If h is not separable, the computation cost of computing $h \cdot I = O(n^2m^2)$, since we need to perform m^2 operations per pixel and the image has n^2 pixels.
If h is separable, we can only do a 1D horizontal and a 1D vertical convolution so that the computation cost of computing $h \cdot I = O(n^2m)$, since we only need to perform 2m operations per pixel.

6 Question 6

Let's create a 3*3 separable filter

A = $\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$

Let's create a 3*3 separable filter

B = $\begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}$

Therefore, A + B = $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$

which is also separable

7 Question 7

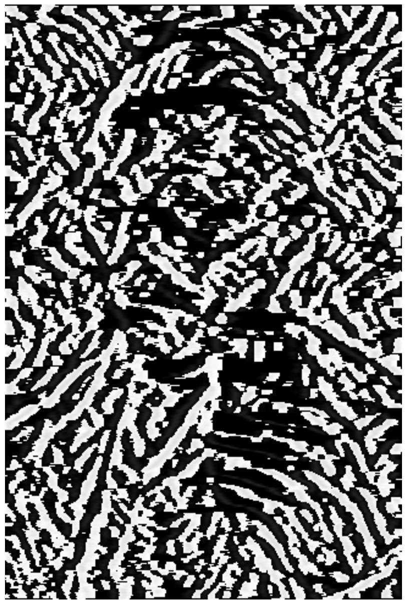


Figure 5: Applied derivative of Gaussian in x and y axis to portrait.jpg

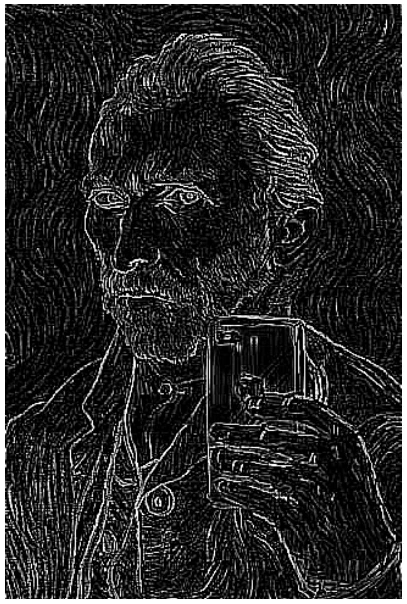


Figure 6: Applied Laplacian of Gaussian to portrait.jpg

The derivation of a Gaussian-blurred input signal is identical to filter the raw input signal with a derivative of the gaussian. In the scipy method gaussian_filter() the parameter order determines whether a derivative of the Gaussian function shall be applied. I generated an image from filtering with derivative of gaussian along x axis and an image along y axis, and also an image along x and y axis together.
To convole with the laplacian of gaussian filter, we simply called the builtin function in OpenCV2.

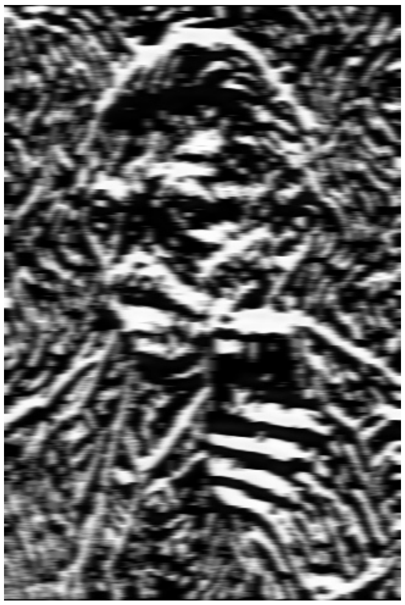


Figure 7: Applied derivative of Gaussian in x axis to portrait.jpg

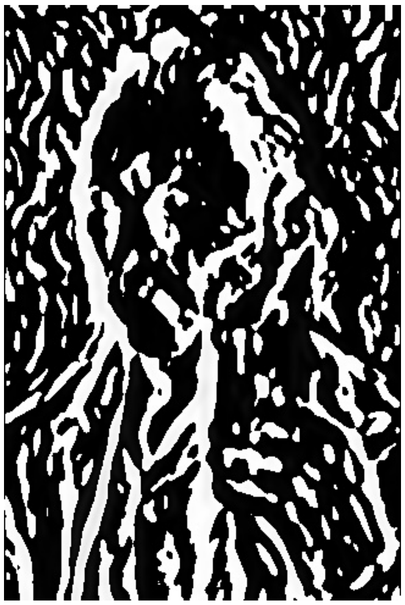
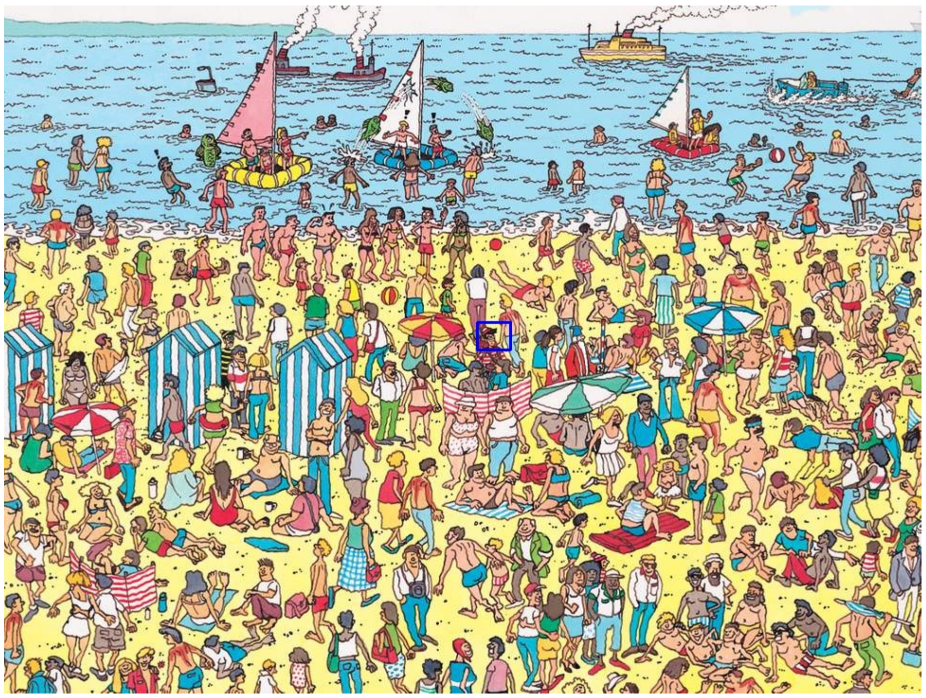


Figure 8: Applied derivative of Gaussian in x axis to portrait.jpg

8 Question 8



9 Question 9

9.1 Filter image with derivative of Gaussian:

Edge detection is prone to noise so that we need to apply a smoothing via Gaussian blur to filter out any noise from the image

9.2 Find magnitude and orientation of gradient:

Gradient magnitudes and directions are calculated at every single point in the image. The magnitude tells whether there lies an edge or not. A high gradient magnitude means the color changes rapidly, implying an edge. The gradient points in the direction of most rapid change in intensity. The gradient direction can be calculated by: $\phi = arctan(\frac{\partial f}{\partial y}/\frac{\partial f}{\partial x})$. The edge strength is given by $\sqrt{(\frac{\partial f}{\partial y})^2 + (\frac{\partial f}{\partial x})^2}$

9.3 Non-maxima suppression:

The purpose for this step is to check if pixel is local maximum along gradient direction. We iterates over all pixels and suppress the ones that are not local maxima along the gradient direction by comparing the magnitude with the upper threshold. By doing so, we can remove any unwanted pixels which may not constitute the edge.

9.4 Linking and thresholding:

From last step, pixels along the edge may not survive the thresholding. We can prevent it by using a high threshold to start edge curves and a low threshold to continue them based on the direction of gradient. Any edges with intensity gradient more than high threshold are sure to be edges and those below low threshold are sure to be non-edges, so discarded. Those who lie between these 2 thresholds are classified edges or non-edges based on their connectivity.

10 Question 10

An edge is a place of rapid change in the image intensity function so the extrema of first derivative of the intensity function can be viewed as the indication of an edge. At the same time, peaks or valleys of the first derivative of the input signal correspond to zero-crossing of the second derivative of the input signal which is Laplacian. We do the edge detection by first smoothing the image using Gaussian filter and then by finding zero-crossing using Laplacian. Since the commutative of convolution operation, this process (Laplacian of Gaussian-filtered image) is equivalent as convolving Laplacian of Gaussian to origin image.

11 Question 11

To get rid of the details from the background, I used the 400 as high threshold, 200 as low threshold.