COMP 6771 Image Processing: Project

Student name: Yunqi Xu

Student id: 40130514

December 7, 2022

1 Review

1.1 Review of Bilateral Filter

Bilateral filter is an angorithm which is nonlinear, edges perserving and Gaussian denoising method proposed by Tomasi in 1998[cite].

The main method that the bilateral filter utilized are two Gaussian filters. One is calcualted based on the Geometric closeness. Another is calculated based on their photometric similarity Below is the process of Bilateral Filtering an image:

Step1: Calculate Domain filter based on the σ_d and k_size, $c(\xi, x) = e^{-\frac{1}{2}(\frac{d(\xi,x)}{\sigma_d})^2}$, where, $d(\xi, x)$ is the Euclidean distance between ξ and x.

Step2: for a certain position (x, y) in an input image, calculate Range filter based on the σ_r, k_size and input image, $s(\xi, x) = e^{-\frac{1}{2}(\frac{\delta(f(\xi),f(x))}{\sigma_r})^2}$, where, $\delta(\phi, f)$ is a suitable measure of distance between the two intensity value ϕ and f.

Step3: Calculate the sum of mutiple between domain filter * range filter with a pitch($size = k\_size * k\_size$) of input image which the center is (x, y) to fet a new value of (x, y).

Step4: Repeat Step 2 and Step 3 until all pixel are visited.

1.2 Review of another paper

2 Details of Bilateral Filter

This section contains three main parts. we will firstly present the result of our re-implement method, and then compared the bilater filter with other baseline algorithm in terms of other low pass filters that usually blur the image but also blur edges. Finally, we will discuss about the Bilateral filter from both advantages and disadvantages and the difficulties during the implement process.

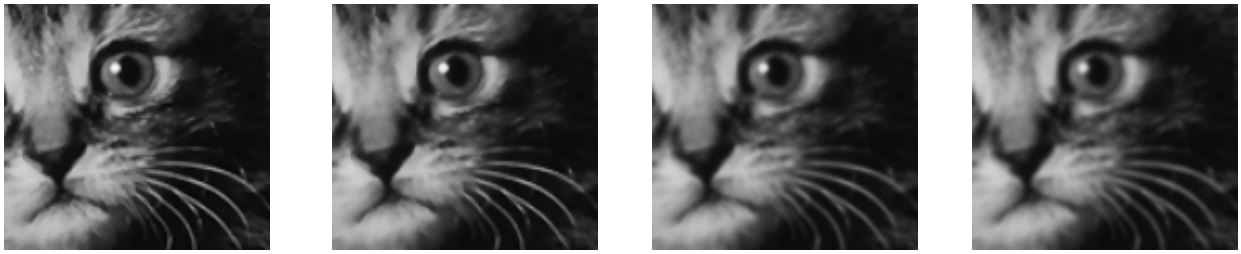
2.1 Result of the Re-implement algorithm

The main purpose for this section is providing an provement of the successful re-implement of the Bilateral filter. During our experiment, we use the images from the Bilateral filter paper[cite here], inputted the same parameters that the paper utilized, and compared our result with the oroginal paper and also the build-in function with Opencv-python offcial to prove the successfully re-implement of our code.

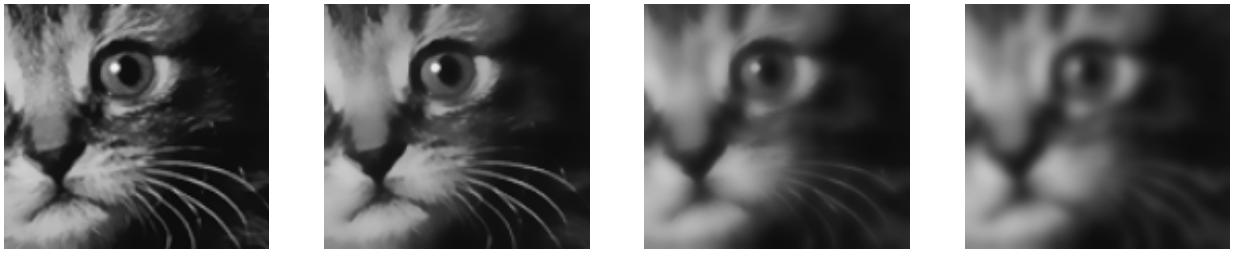
2.1.1 Test result on gray images

Firstly, we present our result compared with the result indicated in the paper, with the same inputted parameters and the same pattern. In the papre, the author use the parameters $\sigma_d = (1, 3, 10)$ and $\sigma_r = (10, 30, 100, 300)$ correspond to each consisting the parameter set. We also use the $kernel\_size = 25$. The Fig. 1 presents the output of our re-implement Bilateral filter. In this

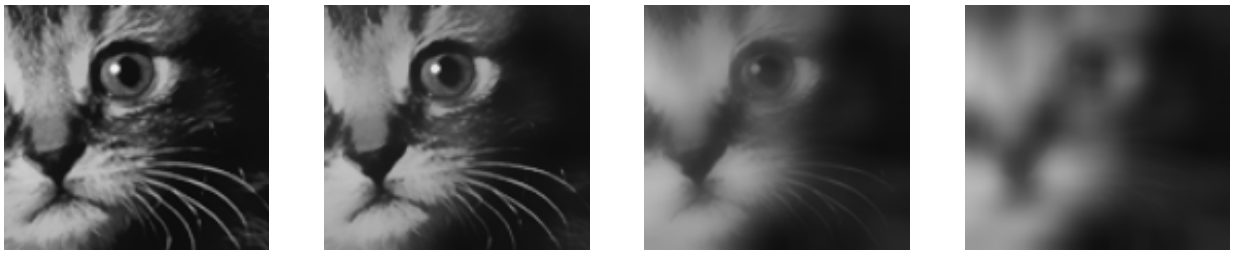
Figure, we can find the blur trend are similar with Figure 3. from the Bilteral filter paper. The Fig. 1(a) has the most clean result, but part of small noises still could be find in it. On the other hand, the Fig. 1(l) is the most blur one in these outputs, it only remain a blurry profile of the cat.



(a) $\sigma_d = 1, \sigma_r = 10$ (b) $\sigma_d = 1, \sigma_r = 30$ (c) $\sigma_d = 1, \sigma_r = 100$ (d) $\sigma_d = 1, \sigma_r = 300$



(e) $\sigma_d = 3, \sigma_r = 10$ (f) $\sigma_d = 3, \sigma_r = 30$ (g) $\sigma_d = 3, \sigma_r = 100$ (h) $\sigma_d = 3, \sigma_r = 300$



(i) $\sigma_d = 10, \sigma_r = 10$ (j) $\sigma_d = 10, \sigma_r = 30$ (k) $\sigma_d = 10, \sigma_r = 100$ (l) $\sigma_d = 10, \sigma_r = 300$

Figure 1: A detail figure with bilateral filters with various range and domain parameter values by implement code

As the comparison, we also make an experiment of input the same parameter pairs in to the build-in opencv-python funtion to explore whether it will have the same output as our code. Fig. 2 prove the results are samilar.

(a) $\sigma_d = 1, \sigma_r = 10$ (b) $\sigma_d = 1, \sigma_r = 30$ (c) $\sigma_d = 1, \sigma_r = 100$ (d) $\sigma_d = 1, \sigma_r = 300$

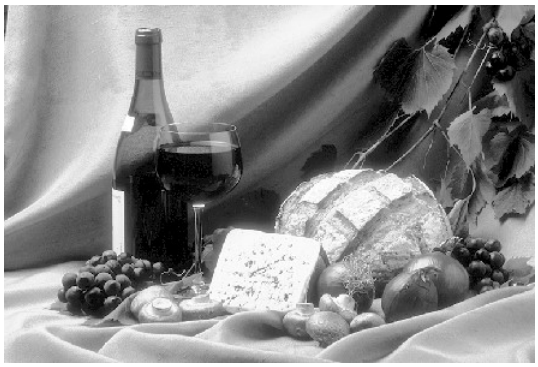
(e) $\sigma_d = 3, \sigma_r = 10$ (f) $\sigma_d = 3, \sigma_r = 30$ (g) $\sigma_d = 3, \sigma_r = 100$ (h) $\sigma_d = 3, \sigma_r = 300$

(i) $\sigma_d = 10, \sigma_r = 10$ (j) $\sigma_d = 10, \sigma_r = 30$ (k) $\sigma_d = 10, \sigma_r = 100$ (l) $\sigma_d = 10, \sigma_r = 300$

Figure 3: A detail figure with bilateral filters with various range and domain parameter values by re-implement code of cat

Fig. 3 are some other outputs use the cat image from paper and the same parameter sets. The same as the figures from the paper, some detils, such as the Kitten's whiskers, also can be remained after filtering.

The Fig. 4 are some other outputs use the figures from Bilateral filter paper. The Fig. 4(a), 4(c) are the oroginal images. As present in Fig. 4(b), the salt and pepper noise can be removed, at the mean time, the edge information can be keeped as shown in Fig. 4(d). In this experiment, we use same parameters which the paper used. $\sigma_d = 3$ and $\sigma_r = 50$.



(a) Original snack

(b) Smoothed snack

(c) Original Onion

(d) Smoothed Onion

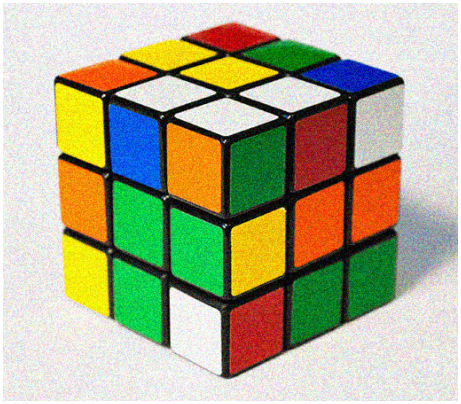
Figure 4: The Bilater filtering result of snacks and the Onion detail

2.1.2 Test result on color images

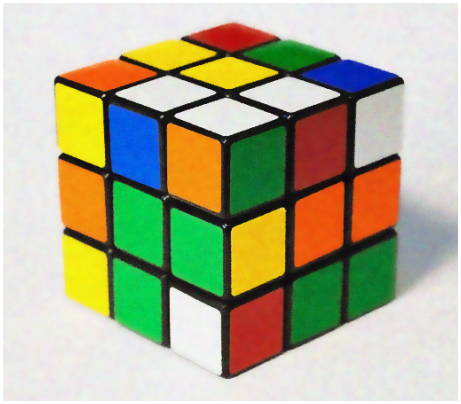
The bilateral filter is not only suitable for filtering gray image, but also achieving successful at filtering color images. Fig. 5 has presents the result of filtering on color images. In these images, the left half (Fig. 5(a), 5(c), 5(e), 5(g))are original imput images. The right half(Fig. 5(b), 5(d), 5(f) and 5(h)) are the result of those color images with Bilater filtering. Compared with filtering gray images, filtering color images need to convert 3-channels RGB images into the CIE-lab space, and then generate two filters based on geometric and photometric information that an image provided. Noises on the original input images could be removed.

(a) Children

(b) Smoothed Children



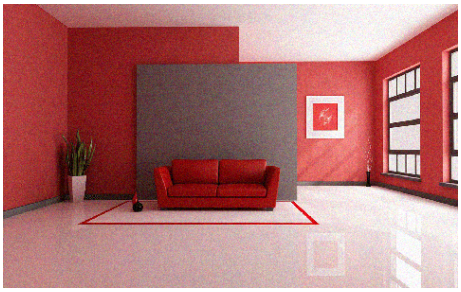
(c) Rubik's cube



(d) Smoothed Rubik's cube

(e) Sky

(f) Smoothed sky



(g) Home

(h) Smoothed home

Figure 5: Outputs of color images

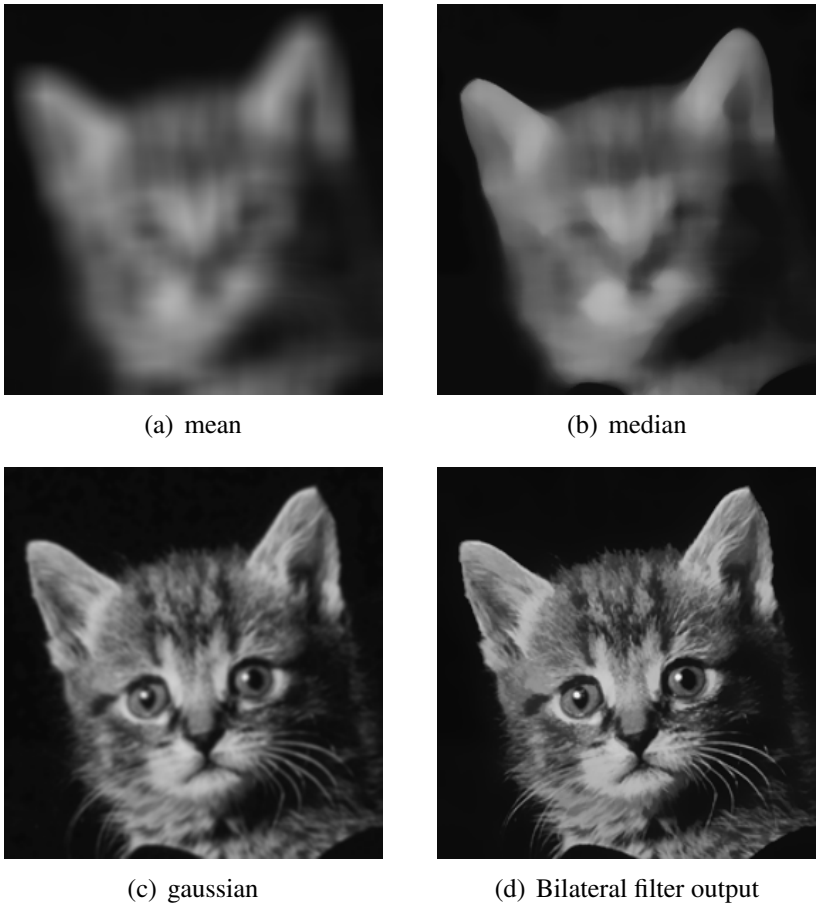
2.2 Compare with other baseline algorithm

In this section, we compared the filter result of Bilateral filter with other baseline algorithm to present the advantages of our code. The baseline low-pass filters we used are mean filtering, median filtering and gaussian filtering. All of this filters are classic filtering in terms of image processing. The Section 2.1 not only prove the result of the re-implement code, but also indicate that the general applicability of Bilateral filter. Our next experiment, we not only compared the output result passed by different filters from human version level, but also calculate the PNSR[cite paper] from mathmatical level. The equation of PNSR has been shwo in Eq. 1:

$$PSNR = 10 \log 10(\frac{I_{Max}^2}{MSE}) \tag{1}$$

$$MSE = \frac{\sum_{M,N}[I_1(m,n) - I_2(M,n)]^2}{M * N} \tag{2}$$

Where, the I_{Max} is the maximum fluctuation in the input images. In our case, the 8-bit integer input image should have $R = 255$. And the I_1 and I_2 are the two images between the noisey one and the filtered one.



(a) mean (b) median

(c) gaussian (d) Bilateral filter output

Figure 6: output compared with the baseline low pass filters

The Fig. 6 has presented the result between three baseline filters and the Bilateral filter. Is is obvious that the the Fig. 6(a) and Fig. 6(b) get a very bad result, since all the details and texture inforamtions are lose. The only remaining parts are the contour in each image. The result of Gaussian filter, Fig. 6(c) can achieve a very closely result compared with Fig. 6(d). However, as we mentioned in Section. 2.1.1, the kitten's whiskers are clean in Fig. 6(d), they are blured in Fig. 6(c). For further prove our statement, we calculate the PSNR between each output image with filter and the original noisy image. The Table. 1 presents the PSNR result. As the result in the table, the PSNR scores of Median filter and Mean filter are too low to use, but the PSNR score of Gaussian filter can achieve a considerablely compititive result as 31.59, which is higher than using Bilateral filter with $\sigma_s = 3$ and $\sigma_d = 100$ or $\sigma_d = 300$ and $\sigma_s = 10$ and $\sigma_d = 30, 100, 300$.

Method	Kernel_size	sigma_s	Sigma_r	PSNR
Bilateral Filter	25	1	10	42.74
			30	32.26
			100	32.59
			300	31.70
Bilateral Filter	25	3	10	40.05
			30	31.37
			100	25.90
			300	24.39
Bilateral Filteral	25	10	10	39.62
			30	29.72
			100	22.38
			300	20.31
Median filter	25	None	None	20.31
Mean filter	25	None	None	19.59
Gaussian filter	25	None	None	31.59

Table 1: The PSNR output of bailteral filter and baseline filters

The Tabel. 1 presents the PSNR reult compared with Bilater filter and other baseline filters.

3 Conclusion

In this section, we will provide the pros and cons of the methods. The methods we used in this report that comes from the lecture. finally, we will provide some brief discription about the difficulties we faced.

3.1 Pros and Cons

Pros:

The Bilateral filter can keep the edge meanwhile blur the noise.

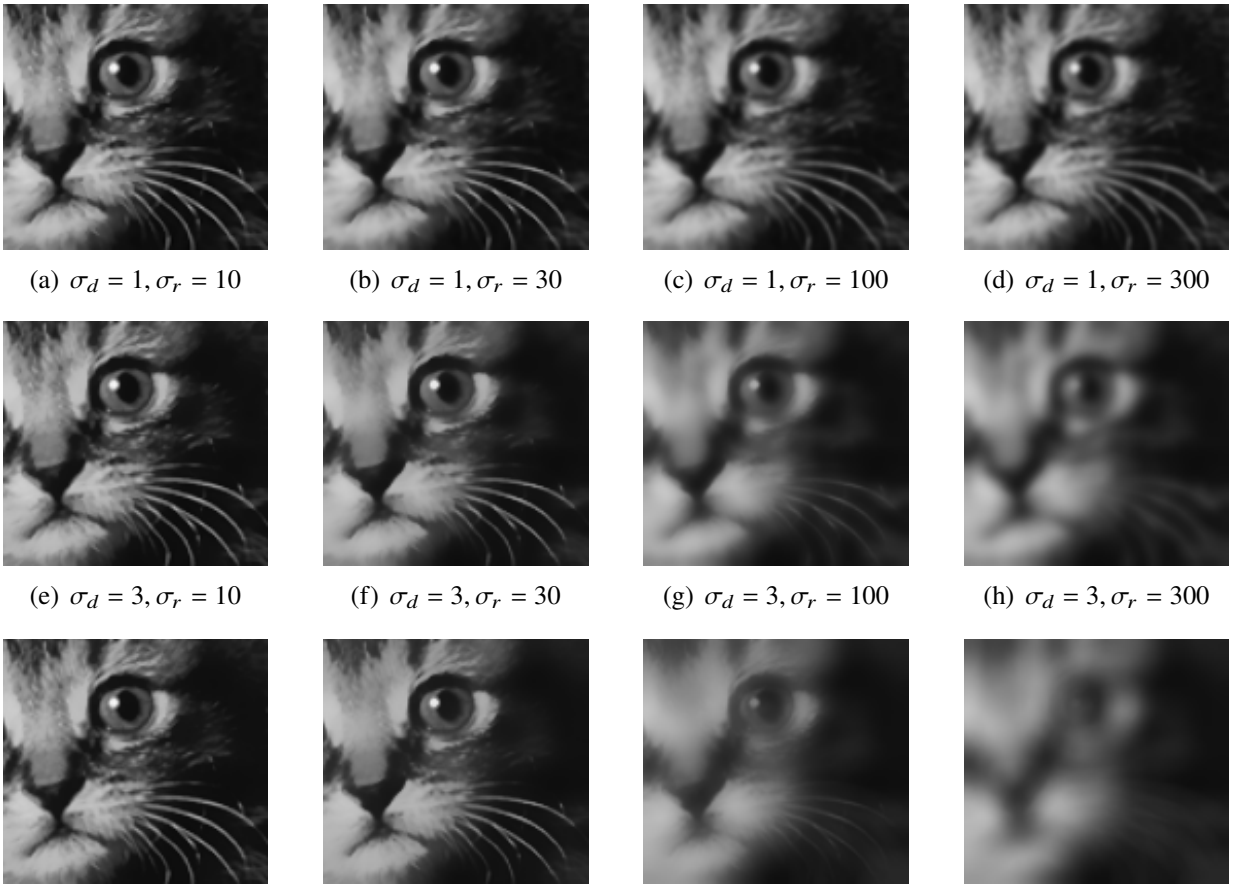
Cons:

When Bilateral neighborhood size gets larger, the Bilateral filtering is slow.

3.2 Method used

3.3 difficulties

The first difficulties is implement the σ_r in the color image. The second is the transform of CIE-lab space.



(a) $\sigma_d = 1, \sigma_r = 10$ (b) $\sigma_d = 1, \sigma_r = 30$ (c) $\sigma_d = 1, \sigma_r = 100$ (d) $\sigma_d = 1, \sigma_r = 300$

(e) $\sigma_d = 3, \sigma_r = 10$ (f) $\sigma_d = 3, \sigma_r = 30$ (g) $\sigma_d = 3, \sigma_r = 100$ (h) $\sigma_d = 3, \sigma_r = 300$

(i) $\sigma_d = 10, \sigma_r = 10$ (j) $\sigma_d = 10, \sigma_r = 30$ (k) $\sigma_d = 10, \sigma_r = 100$ (l) $\sigma_d = 10, \sigma_r = 300$

Figure 2: A detail figure with bilateral filters with various range and domain parameter values by Opencv python