Computational Photography Assignment 3

AndrewID: ranzhang

October 20, 2020

* All the images I tested are included in the data folder. I only included some representative images in this document.

1 Bilateral Filtering

I generated seven sets of images using four techniques. The parameters I used are shown below.

Set Number	σ^s	σ^r
1	1	0.05
2	3	0.05
3	6	0.001
4	6	0.05
5	6	0.1
6	6	0.15
7	12	0.001

Table 1: Bilateral Filtering Settings

According to the paper, when using bilateral filter, the σ^r could be set to 0.1% of the total range of color values. So I also tested this value. Besides, I filtered flash image with the same settings when generating the F^{Base} at the beginning, but it turned out that the merged A^{Detail} images are blurry especially when σ^s is large. So I adjusted the σ when bilateral filtering the flash image, where σ^s equals 1 and σ^r equals 0.05.

For the basic bilateral filtering, $\sigma^s = 6$ could generate good images. When $\sigma^r = 0.05$, the details on the wheat are well kept, but the background is noisy. When $\sigma^r = 0.15$, the situation is opposite. So I think $\sigma^r = 0.1$ is appropriate to get a good result.

Figure 1: Basic Bilateral Filtering Comparison
(Left to right: $\sigma^s = 6$ and $\sigma^r = 0.05$,$\sigma^s = 6$ and $\sigma^r = 0.1$,$\sigma^s = 6$ and $\sigma^r = 0.15$)

For the joint bilateral filter, the $\sigma^s = 6$ could still work well. If σ^s is too small, the noise is not removed. If σ^s is too large, the image is totally blurred. As the paper mentioned, when the $\sigma^s = 6$, $\sigma^r = 0.001$ could

get a good result.The noise on the background is removed while the detail on the wheat is still kept. But the shadows between the wheat are removed as well. The reason is, when merging with the flash image, these shadows are lit by flash.

Figure 2: Joint Bilateral Filtering Comparison
(Left to right: $\sigma^s = 6$ and $\sigma^r = 0.001$,$\sigma^s = 6$ and $\sigma^r = 0.05$,$\sigma^s = 12$ and $\sigma^r = 0.001$)

Then I compared images filtered by basic method and joint method and their difference images. The differences are mainly on the specular spots on the pots and the wheat. When σ^r is large (0.1), the joint filtered image is more blurry than the basic filtered image especially on these areas. But the noise is removed similarly. When the σ^r is small (0.001), the difference between the wheat and specular areas is less, which means the joint filtered image is less blurry. Meanwhile, the basic filtered image shows much more noise than the joint filtered image.

Figure 3: Basic And Joint Bilateral Filtering Comparison, $\sigma^s = 6$ and $\sigma^r = 0.1$
(Left to right: Basic, Joint, Difference)

Figure 4: Basic And Joint Bilateral Filtering Comparison, $\sigma^s = 6$ and $\sigma^r = 0.001$
(Left to right: Basic, Joint, Difference)

Then I compared the A^{Detail} and A^{NR}. These two sets of images are very similar. The main differences are the wheat and some areas on the pots, but not limited to the specular areas since these areas have more details than the others. As the σ^r decreases, the images differ more.

Finally, I compared the masked images and detail filtered images. The differences between these two sets of images are the outline of the wheat lamp and pots, together with the specular areas on the pot. Such difference indicates the function of the mask which removes the shadows and specularities created by the flash. But as the σ^r decreases, there are also many noises added back. The reason is that the basic filtered image is added in these masked areas, where the noise could not be removed by a small σ^r value (0.001).

The best settings for these four bilateral filtering methods are shown in Table 2. For the basic bilateral

Figure 5: Joint And Detail Bilateral Filtering Comparison, $\sigma^s = 6$ and $\sigma^r = 0.1$
(Left to right: Joint, Detail, Difference multiplied by 5)

Figure 6: Joint And Detail Bilateral Filtering Comparison, $\sigma^s = 6$ and $\sigma^r = 0.001$
(Left to right: Joint, Detail, Difference multiplied by 5)

Figure 7: Detail and Mask Bilateral Filtering Comparison, $\sigma^s = 6$ and $\sigma^r = 0.1$
(Left to right: Detail, Mask, Difference multiplied by 5)

Figure 8: Detail And Mask Bilateral Filtering Comparison, $\sigma^s = 6$ and $\sigma^r = 0.001$
(Left to right: Detail, Mask, Difference multiplied by 5)

filter, the advantages are that it needs less calculation, no need for flash images and therefore, gets away from the shadows and specularities caused by flash. The disadvantage is that it needs to balance between denoising and keeping details. For the joint bilateral filter, the advantage is that it still needs less calculation even flash images are needed. For the disadvantages, the shadows and specularities are included. Also, it may lose some details and textures compared with detail bilateral filter. For the detail bilateral filter, the advantage is that some textures are kept. But the disadvantage is that it needs to apply bilateral filter onto the flash image, which needs calculation and more parameters to be adjusted. For the mask filter, the advantage is that shadows and specularities are removed while the textures are kept. But the disadvantage is that noises in the shadows and specular areas caused by flash are added back due to the usage of basic bilateral filtered images with small $sigma_r$. Besides, generating the mask could be a tricky task.

Method	σ^s	σ^r
Basic	6	0.05
Joint	6	0.001
Detail	6	0.001
Mask	6	0.001

Table 2: Bilateral Filtering Methods Best Settings

2 Gradient-domain processing

2.1 Differentiate and then re-integrate an image

This is the image I generated. There are differences between the source image and my image because I calculated the divergence of the image by using kernel [1, -1] twice instead of [1, -2, 1].

Figure 9: Differentiate and then re-integrate an image, N=5000, $\epsilon = 0.001$
(Left to right: original image, reintegrated image, difference)

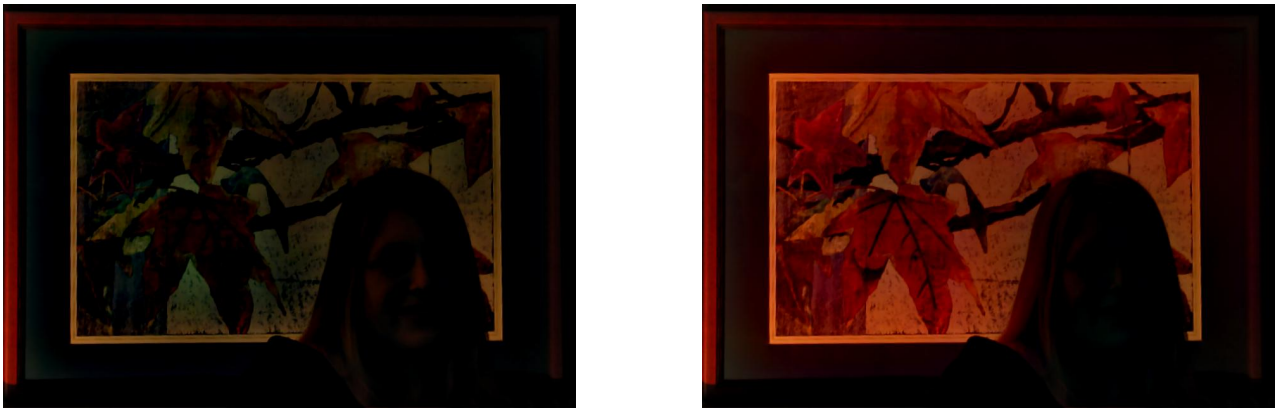

Figure 10: Differentiate and then re-integrate an image (Left to right: N = 100, $\epsilon = 0.1$)

I also tested different N and convergence ϵ. When the N is not big enough, the image would first generate the edges. The textures inside are missing. When the ϵ is big, the image would show dimmer.

2.2 Create the fused gradient field

For this part, I set N to be 5000 and ϵ to be 0.001. The iterations usually end when N is about 1000. When I adjusted σ, the images are visually identical and the iteration numbers are the same. The difference image indicates that the paths of gradient decent defer.

Figure 11: Gradient fused images comparison $\tau_s = 0.8$
(Left to right: $\sigma = 30$,$\sigma = 40$,Difference multiplied by 10)

But when I adjusted the τ_s, the images are visually different. As the τ_s increases, the specular area is less obvious but the area around it is brighter as a whole. (These three images shown in this document seem similar, but the original files included in the data folder are more different)

Figure 12: Gradient fused images comparison $\sigma = 40$ (Left to right: $\tau_s = 0.8$,$\tau_s = 0.9$,$\tau_s = 0.95$)

Figure 13: Gradient fields $\sigma = 40, \tau_s = 0.9$ (Left to right: $\nabla\alpha, \nabla\Phi', \nabla\Phi^*$)

I tested different initialization of I^*. The four images are visually identical. The stop iteration numbers are the same. Even their difference images are all black. They only have slight difference around a few columns and rows on the edges of the images.

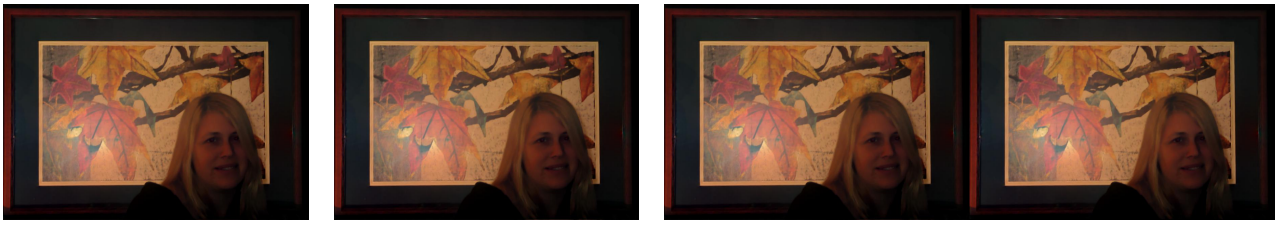

Figure 14: Initialization comparison $\sigma = 40, \tau_s = 0.9$ (Left to right: ambient, flash, average, zero)

3 Capture your own flash / no flash images

3.1 Bilateral Filter

To generate the ambient image with more noise, I set the ISO to be 25000. So, I used a larger space kernel for this image. Then I tested different parameters to generate the fused image.

Figure 15: Source images captured by myself(Left to right: ambient, flash)

Figure 16: A^{Detail} and final result before being optimized ($\sigma_s = 12, \sigma_r = 0.001$)

Like I mentioned earlier, masked filtering does reduce the noise and keep more details, but it introduces the noise in base ambient image since the σ_r is very small. My solution is that I adjusted the τ_s and iteration number of erosion and dilation when generating the mask to make it as precise as possible. Also, when I merge the base ambient image, I used one with larger σ_r (0.15 instead 0.001) to avoid introducing more noise.

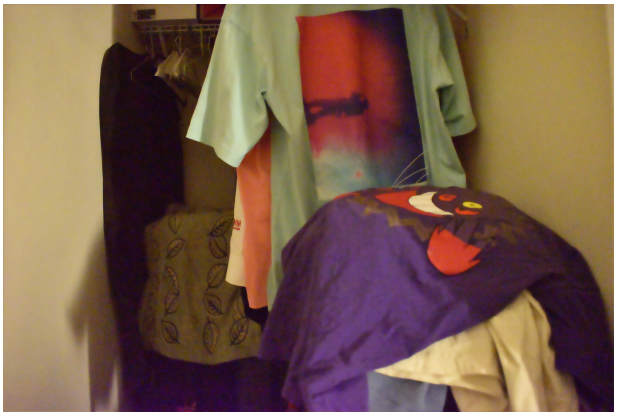
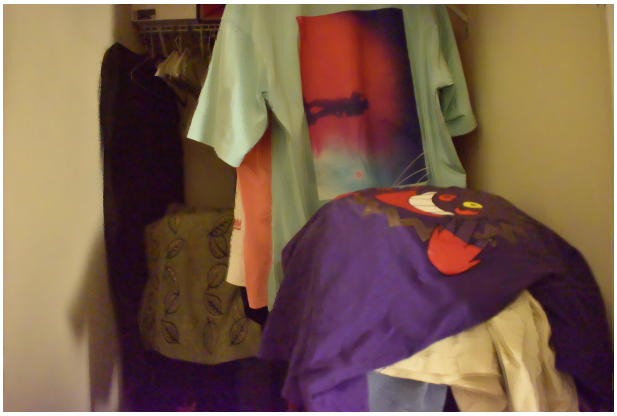

Figure 17: A^{Base} used for replacing the shadow and specular areas only ($\sigma_s = 6, \sigma_r = 0.15$) and final result

3.2 Gradient fused image

I first used similar parameters ($\sigma = 40, \tau_s = 0.9$) but the specularity is not removed at all. Then I decreased the τ_s and got better result. But the generated image is green. The reason might be, unlike the referred images, the images I took has different lighting colors since the lamp in my room is yellow instead of white. So I tried to apply white balance on it. Some part of it seems better but the overall image is blue.

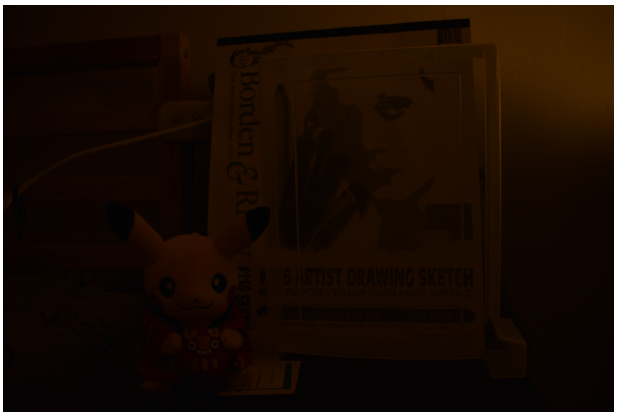
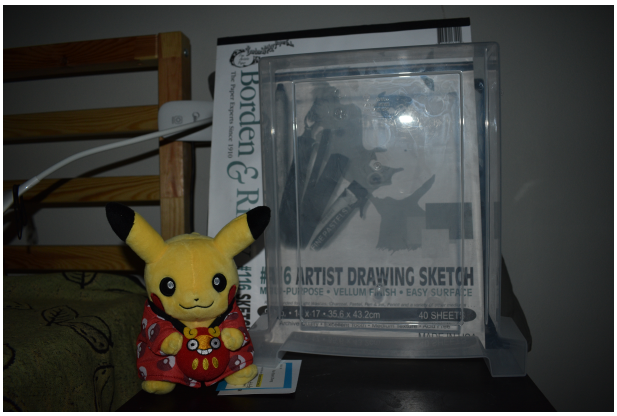

Figure 18: Source images captured by myself (Left to right: ambient, flash)

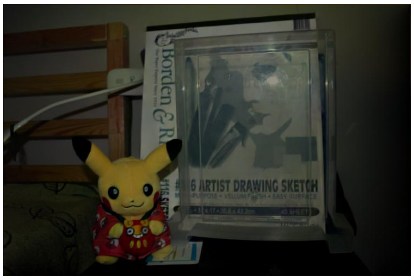
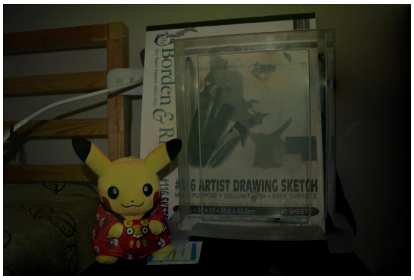
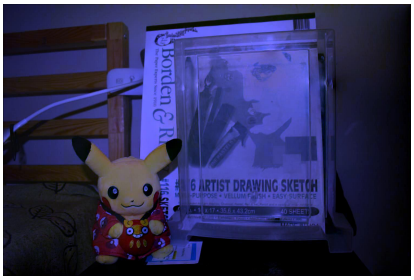

Figure 19: Fused images $\sigma = 40$ (Left to right: $\tau_s = 0.8, \tau_s = 0.5, \tau_s = 0.5$ white balanced)

4 Reflection removal in flash / no-flash photography

I started with the parameters mentioned in the paper that $\sigma = 40$ and $\tau_{ue} = 0.1$. But the indoor objects are not shown correctly. Then I increased the τ_{ue}. The indoor objects are brighter but the outdoor scene is darker. The reason might be, I did not shot a very bright outdoor scene, as what the referred paper did. So

Figure 20: Source images captured by myself (Left to right: ambient, flash)

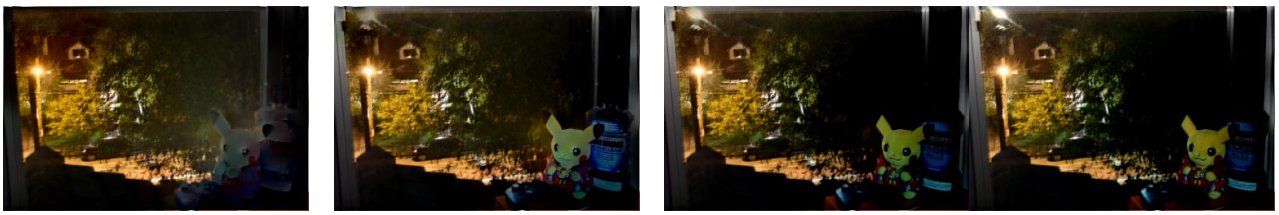

Figure 21: Reflection removal tests, $\sigma = 40$ (Left to right: $\tau_{ue} = 0.1$,$\tau_{ue} = 0.3$,$\tau_{ue} = 0.4$,$\tau_{ue} = 0.5$)

Figure 22: Reflection removal result, $\sigma = 40$ $\tau_{ue} = 0.4$)

the areas near the indoor objects are dark, where the algorithm has no information to recover the scene after the reflection is removed. Finally I picked $\tau_{ue} = 0.4$ and generated the final image.

5 Implement acceleration techniques for Poisson solver

5.1 Implement Jacobi preconditioning

I implemented the Jacobi preconditioning algorithm. When I calculated the Jacobi diagonal preconditioner $diag(L)$, I realized that such preconditioner is a matrix with value -4 on the diagonal line only. Then I appilied this preconditioner by multiply the whole image with -0.25, which makes no difference to the gradient descent. The result is exactly the same as CGD.

5.2 Implement a simple multi-grid approach

I implemented this simplified multi-grid approach and tested it.

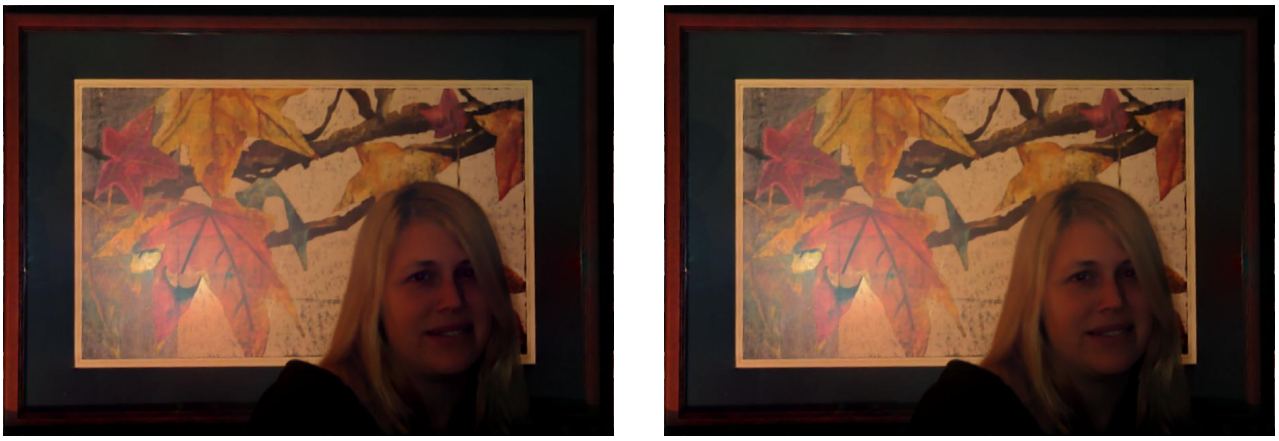

Figure 23: Fused image with multi-grid method (Left to right: $\epsilon = 0.01$,$\epsilon = 0.001$)

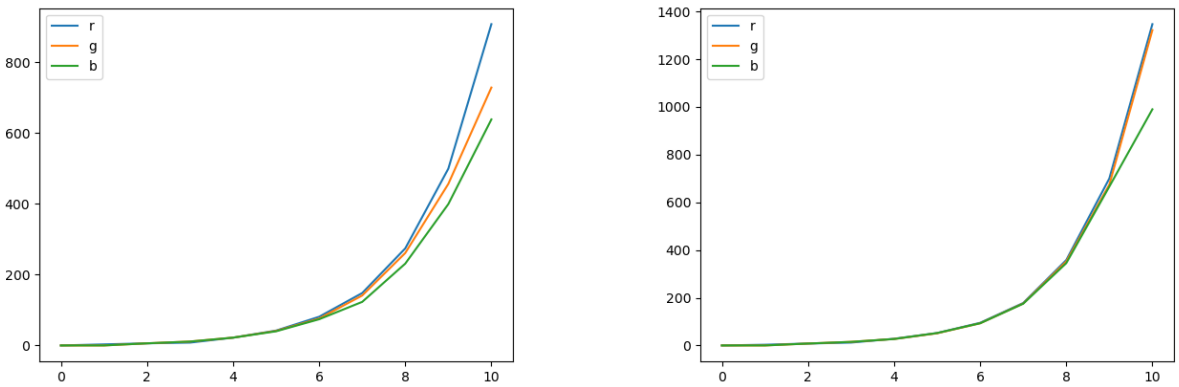

Figure 24: Iteration numbers in Gaussian pyramids (Left to right: $\epsilon = 0.01$,$\epsilon = 0.001$)

The generated images are almost the same as CGD since I set N to be very large so the Poisson Solver always stops after reaching the ϵ. But the numbers of iterations different methods take are different. For the multi-grid method, the iteration number increases exponentially as the grid size increases. When the ϵ is 0.01, the max grid takes 907, 728, 638 iterations each channel but the CGD takes 907, 826, 870 iterations

respectively. When the ϵ is 0.001, the max grid takes 1347, 1322, 990 iterations each channel but the CGD takes 1235, 1214, 1163 iterations respectively. So when the ϵ is larger, multi-grid method is faster than CGD. When the ϵ decreases, the CGD becomes comparatively faster.