Computer Vision II – Homework Assignment 3
Solution from Group 28

Jiahui Shi, Fan Yang, Zhu Yang

June 21, 2023

Problem 1 – Graphcuts and α-expansion for Image Denoising

1-6. For solutions to programming exercises, please see the A3_problem1.py file.

Figure 1: comparison of psnr in the worse and best case after processing

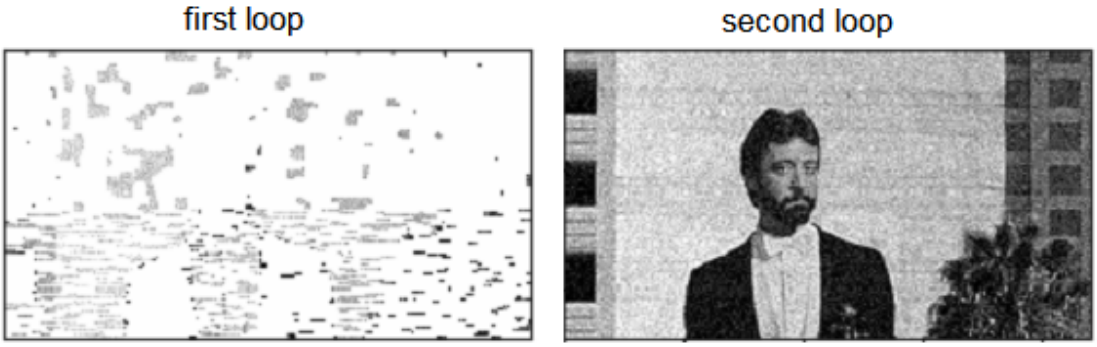

Figure 2: comparison of psnr after first loop and second loop processing

7. Discussion of results: The basic idea is to use the noise image and generated image with all same pixel values from 0 to 255 as two references, in order to update the initial image. Theoretically when the parameters are well set the denoised image (initial image after updating) should have a higher psnr than noise image.
In the beginning we only let the program run one turn, which means only begin with one for loop to choose the second reference pixel value to be 0 to 255.
We tried a few sets, Most of them are worse cases. (fig. 1 (left)). Among them, $psnr\_after$ are always lower than $psnr\_before$.

Then we let the for program run many turns using a while loop, to make sure higher $psnr\_before$ can be obtained. But we also try many parameter sets, even run seven turns for about three hours, the best results is to rebuild the noise image, which means $psnr\_after$ is the same as $psnr\_before$. (fig. 1 (right))

So we decide to check $psnr\_after$ in each iterate of candidate value. If the higher $psnr\_after$ is obtained, then break the candidate value for loop and output results.
The parameter set is $\sigma = 5, \lambda = 5$. The result after first full loop run(0 to 255) is shown in (fig. 2 (left)). The $psnr\_before$ and $psnr\_afterresults$ are 24.66354342728765 and 5.938440787442377 respectively. After the second iteration in the second loop run, a higher $psnr\_after$ is obtained: higher psnr result is obtained: 24.66354342728765 24.673363747996653. (fig. 2 (right)). The difference is not big so the image denoised is very similar to image noise.

Problem 2 – Horn-Schunck optical flow using PyTorch

1-7. For solutions to programming exercises, please see the A3_problem2.py file.

5. By visualization, we got the images of I_1, I_2^w and the difference between both images(fig. 3). By observing the resulting image, it can be seen that because the warp has played a role, the difference becomes smaller and the edge of the main object is more accurate. This is because warp remaps img2 to im1 through warp transformation, so that the main objects at time t and t+1 coincide.

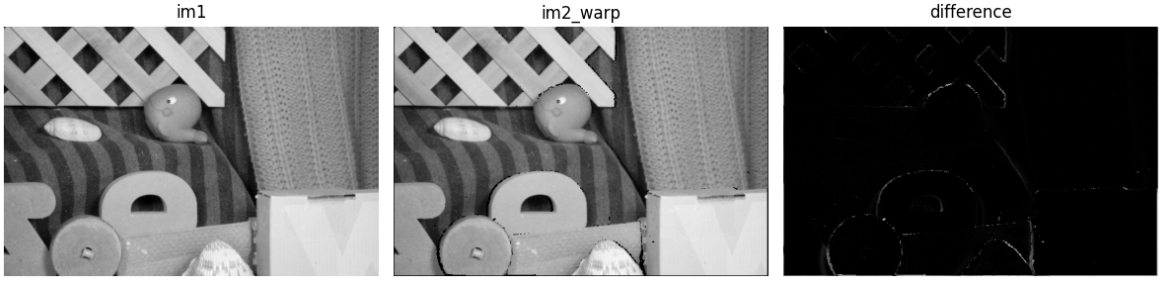

Figure 3: Visualization of I_1, I_2^w and the difference

7. Discussion of results: As can be seen from the picture of the observation result, this result is highly similar to the expected result in the question. Only the saturation is slightly lower. We speculate that this is due to the convolution kernel. The convolution kernel can be further tuned to produce better results. Therefore, we made a symmetrical transformation of the convolution kernel and obtained an optimized result with better saturation.

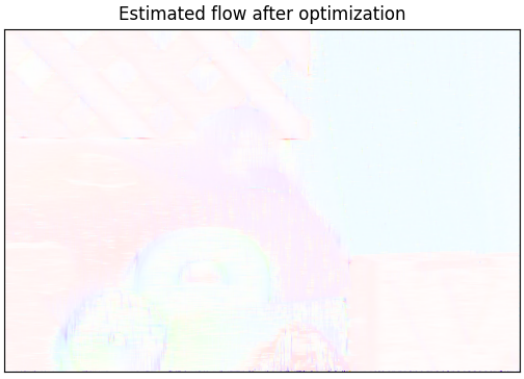

Figure 4: flow estimate result from a plain gradient descent optimizer

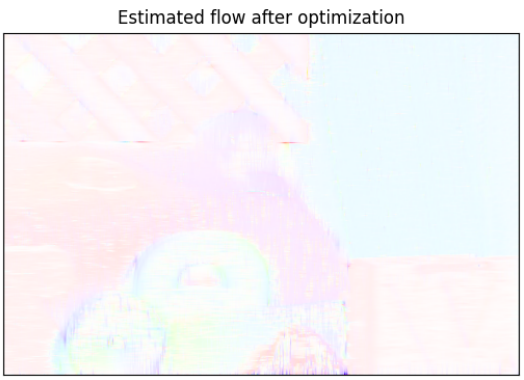

Figure 5: flow estimate result from a plain gradient descent optimizer with better saturation