National Tsing Hua University

Department of Electrical Engineering

EE6620 Computational Photography, Spring 2022

Homework #2

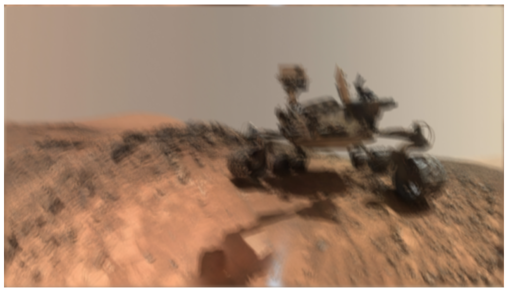
Non-Blind Deblurring

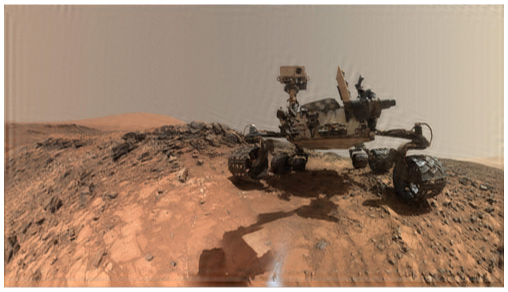
Assigned on April 1, 2022

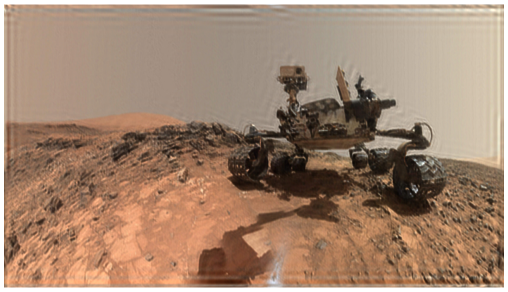
Due by April 22, 2022

Homework Description

A slow shutter speed will introduce blurred images due to camera shake. The objective of this assignment is to implement several non-blind deblurring algorithms and analyze the effects on blurred images. In part 1-3, you will implement Wiener deconvolution, Richarson-Lucy (RL) deconvolution [1] [2] and its bilateral variant (BRL) [3]. And in part 4, you are going to solve deblurring problem in total variation model [4]. There are two synthetic images in this assignment. The **Curiosity-small** and **Curiosity-meidium** blurred images were synthesized using small and medium blur kernel sizes respectively. For simplicity, the blurring was performed in non-linear ($R'/G'/B'$) domain.


Blurred


Wiener deconv


RL deconv

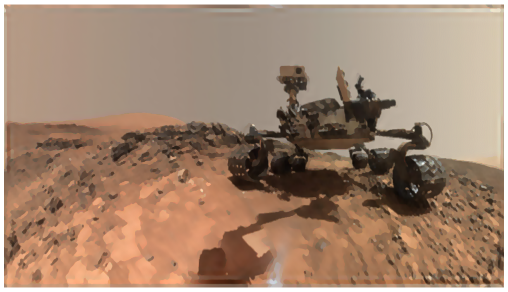

BRL deconv

Table 1: Scores

Items		Score
Implementation (50%)	Wiener deconvolution	5%
	RL deconvolution	15%
	BRL deconvolution	20%
	Total variation regularization	10%
Report (50%)	Exploration	20%
	Research study	20%
	Free Study	10%

1 Implementation (50%)

In this section, you will implement several algorithms for non-blind deblur flows. For each flow, you need to establish in the python codes (HW2/code/deblur_functions.py or HW2/code/TV_functions.py), and coresponding test functions are provided in (HW2/code/test_deblur_functions.py). In test_deblur_functions.py, an argument $TEST\_PAT\_SIZE$ can be assigned to either small or large. To speed up development, you could set $TEST\_PAT\_SIZE$ to small for quick debugging; **to get the full score in this section, you need to pass test functions with** large **setting. And please do not modify original function I/O. Otherwise, your implementation will be considered as incorrect.**

There are some details you should comply with in implementation:

1. Perform Wiener, RL and BRL deconvolution for the **three color channels separately**.
2. Use **floating-point** operations and normalize the array element (e.g., B, I, K) to **[0, 1]** before deblurring.
3. Normalize the blur kernel K such that $\sum k_i = 1$, before using it.
4. For doing convolution and padding, the **symmetric boarder condition** (Appendix: Figure 3) is used in reference answer. However, you can try some different boundary conditions in your report.
5. RL/BRL can work for different choices of initial images $I^{(0)}$. The blurred image (original image) is a typical choice, but you can try other initial images (for example, a flat black image) in your report.
6. When doing pixel-wise division in RL and BRL deconvolution (i.e., x/y), **code it as x/(y+DBL_MIN) to avoid zero-division issue.**
7. It is allowed to use following library functions to enhence operating efficiency in this assignment.
   i scipy.signal.convolve2d()
   ii numpy-array operation (e.g., np.exp(), np.sum(), np.fft.rfft2(), np.fft.irfft2(), etc)

1.1 Wiener deconvolution (5%)

Wiener deconvolution can be express as

$$I(\hat{f}) = \frac{B(f)}{K(f)}\left[\frac{|K(f)|^2}{|K(f)|^2 + \frac{1}{SNR(f)}}\right] \tag{1}$$

$$\hat{I} = IFFT\{I(\hat{f})\} \tag{2}$$

where

$$B(f) = FFT\{B\}, K(f) = FFT\{K\} \tag{3}$$

Finish the function Wiener_deconv() in deblur_functions.py. The PSNR should be larger than 60 dB for Wiener_deconv(). To get the correct result, please notice that there are some preprocessing steps (Figure 1) before applying DFT to the blur kernel.

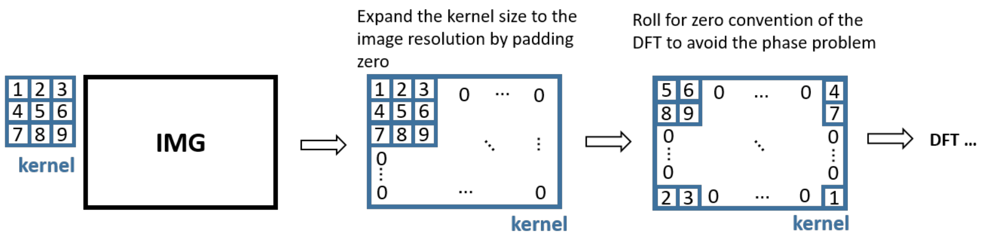

kernel
IMG
Expand the kernel size to the image resolution by padding zero
kernel
Roll for zero convention of the DFT to avoid the phase problem
kernel
DFT ...

Figure 1: Preprocessing of 2D-DFT on blur kernel

To get the full score, please make sure you pass the Wiener deconvolution test function with large $TEST\_PAT\_SIZE$ setting.

a. Wiener deconvolution on curiosity_medium, with $SNR(f)$ = 100.0.
   (tested by test_Wiener_deconv())

1.2 RL deconvolution (15%)

RL deconvolution can be express as

$$I^{t+1} = I^t \circ [K^* \otimes \frac{B}{I^t \otimes K}], \tag{4}$$

where

$$K^*(i,j) = K(-i,-j). \tag{5}$$

Energy function of RL can be formulated as

$$E(I) = \sum_{ch}\left\{\sum((I \otimes K) - B \circ \ln(I \otimes K))\right\}. \tag{6}$$

Finish the function RL() and RL_energy() in deblur_functions.py. RL() function should be able to process the input image by RL deconvolution and RL_energy() function should be able to calculate the energy of the deblurred results of RL. The PSNR should be larger than 60 dB for RL() and the error for RL_energy() should be smaller than 0.05%.

To get the full score, please make sure you pass the following RL test functions with large $TEST\_PAT\_SIZE$ setting.

a. RL on curiosity_small, with iteration = 30.
   (tested by test_RL_a())
b. RL on curiosity_medium, with iteration = 45.
   (tested by test_RL_b())
c. Calculate RL energy function.
   (tested by test_RL_energy())

1.3 BRL deconvolution (20%)

BRL deconvolution can be express as

$$I^{t+1} = \frac{I^t}{1 + \lambda \cdot \nabla E_B(I^t)} \circ (K^* \otimes \frac{B}{I^t \otimes K}), \tag{7}$$

where

$$K^*(i,j) = K(-i,-j), \tag{8}$$

and

$$\nabla E_B(I^t) = 2 \cdot \sum_{y \in \Omega} (\exp(-\frac{|x-y|^2}{2\sigma_s}) \cdot \exp(-\frac{|I(x)-I(y)|^2}{2\sigma_r}) \cdot \frac{(I(x)-I(y))}{\sigma_r}). \tag{9}$$

In (9), x is the center pixel and y are nearby pixels in Ω.
Energy function of BRL can be formulated as

$$E(I) = \sum_{ch} \left\{ \sum \{(I \otimes K) - B \circ \ln(I \otimes K)\} + \lambda E_B(I) \right\}, \tag{10}$$

where

$$E_B(I) = \sum_{x} \sum_{y \in \Omega} \exp(-\frac{|x-y|^2}{2\sigma_s})(1 - \exp(-\frac{|I(x)-I(y)|^2}{2\sigma_r})) \tag{11}$$

Finish the function BRL() and BRL_energy() in deblur_functions.py. BRL() function should be able to process the input image by BRL deconvolution and BRL_energy() function should be able to calculate the energy of the deblurred results of BRL.

The PSNR should be larger than 55 dB for BRL() and the error for energy_BRL() should be smaller than 0.05% .

To get the full score, please make sure you pass the following BRL test functions with large $TEST\_PAT\_SIZE$ setting.

a. BRL on curiosity_small, with
   $\lambda = 0.03/255$, iteration=15, $\sigma_r = 50/255^2$, $r_K = 6$, $r_\Omega = 0.5r_K$, $\sigma_s = (r_\Omega/3)^2$.
   (tested by test_BRL_a())

b. BRL on curiosity_small, with
   $\lambda = 0.06/255$, iteration=15, $\sigma_r = 50/255^2$, $r_K = 6$, $r_\Omega = 0.5r_K$, $\sigma_s = (r_\Omega/3)^2$.
   (tested by test_BRL_b())
c. BRL on curiosity_medium, with
   $\lambda = 0.001/255$, iteration=40, $\sigma_r = 25/255^2$, $r_K = 12$, $r_\Omega = 0.5r_K$, $\sigma_s = (r_\Omega/3)^2$.
   (tested by test_BRL_c())
d. BRL on curiosity_medium,with
   $\lambda = 0.006/255$, iteration=40, $\sigma_r = 25/255^2$, $r_K = 12$, $r_\Omega = 0.5r_K$, $\sigma_s = (r_\Omega/3)^2$.
   (tested by test_BRL_d())
e. Calculate BRL energy function.
   (tested by test_BRL_energy())

1.4 Solving deblurring problem by total variation regularization (10%)

You are going to implement the Total Variation algorithm using ProxImaL[4] in TV_functions.py. A example usage in function TVL1() shows how to implement Total Variation model in norm 1 using ProxImaL. In TVL2(), you need to change the data term from norm 1 to norm 2. In TVpoisson(), you need to change the data term from norm 1 to model the noise in the blurring process as Poisson distribution (hint: use ProxImaL function poisson_norm()).
To get the full score, please make sure you pass the following total variation test functions with large $TEST\_PAT\_SIZE$ setting.

a. TVL2 on curiosity_medium, with $\lambda = 0.01$, iteration=1000.
   (tested by test_TVL2())
b. TVpoisson on curiosity_medium, with $\lambda = 0.01$, iteration=1000.
   (tested by test_TVpoisson())

2 Report(50%)

There are three parts of problems: Exploration, Research study and Free study. Answer the problems clearly and integrally in your report.

2.1 Exploration (20%)

a. A conceptual figure (Figure 2) shows how to evaluate your deblurred result by 1-D DFT scanline. Try to plot a same figure for the deblurred results and a non-blurred image data/curiosity.png. How do you plot the figure? What is the meaning of the figure? How this figure can help you to explain the result?

Do you apply any additional steps? Why? **(5%)**

b. Deblur your own blurred images. Photograph two blurred images with different kinds of kernel. Estimate their blur kernels by yourself, and generate your own deblurred images using BRL deconvolution. A reference method to photograph a blur image with a blur kernel is shown in (Appendix: Figure 4). Provide at least the following information in your report. **Notice that when conducting deconvolution to a photo taken in real life, you should transform the photo to the linear domain first (e.g., $img^{2.2}$), and then transform it back to the nonlinear domain (e.g., $img^{\frac{1}{2.2}}$) after finishing your deblur flow! (15%)**

i Your blurred images, kernels and result deblurred images

ii How do you extract the kernels?

Do you conduct extra preprocess on your blur kernels or images? Why?

iii How do you choose your parameters? (e.g., λ, iteration, etc)

iv Compare the deblur effect for the two kernels.

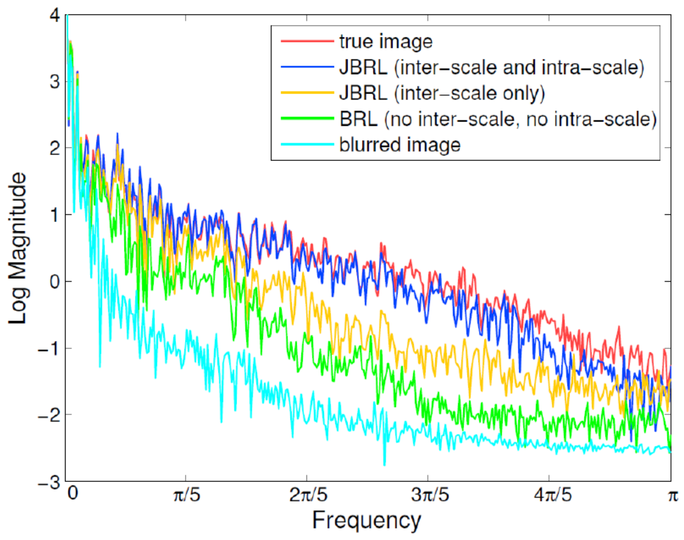


Figure 2: DFT curves of a 1-D scanline

2.2 Research Study (20%)

In this section, you should answer the problem by the two-step research flow: **Assumption** and **Justification**. Based on your observation, propose an assumption, and justify your idea by experiments. This reseaerch flow may be a loop:

observation $\rightarrow$ propose assumption $\rightarrow$ justification $\rightarrow$ modify assumption $\rightarrow$ justification ...

After reaching a satisfactory result, you need to describe your assumption and how you justify it by experiments in the report. Grading criteria for this part are based on the clarity of the report and the rigorousness of the justification.

a. Compare the BRL results of curiosity-medium with different parameters. There are many settings you can vary like λ, σ_r, r_Ω to r_K ratio, padding border condition of convolution, etc. How do these settings influence the result BRL images? Please pick at least 2 settings to do the two-step research flow. **(10%)**

b. In this assignment, you have done many kinds of non-blind deblurring flows (i.e., Wiener, RL, BRL, and TV). And since it is inevitable to have some noise in the picture, being robust to the noisy image is crucial. Which deblur flow is more robust to the noisy blurred image? Propose your assumption and reasons, and then justify your ideas by experiments. **(10%)**

2.3 Free Study (10%)

In this section, please propose **one** interesting topic, and answer it by the two-step research flow. You can also choose your free study topic from the following problems. **TA will consider both the breadth and depth of the free study, choosing the top two students. And 2% and 1% bonus scores on semester will be given to the best and the second-best students**

a. Implement another deblurring flow, and compare the result with the already have.
b. Try to speed up BRL and report the execution time difference.
c. Use energy function to estimate the convergence of RL and BRL.
d. For Wiener deconvolution, establish your flow to find the suitable SNR(f) value of new blurred images.

3 Deliverable

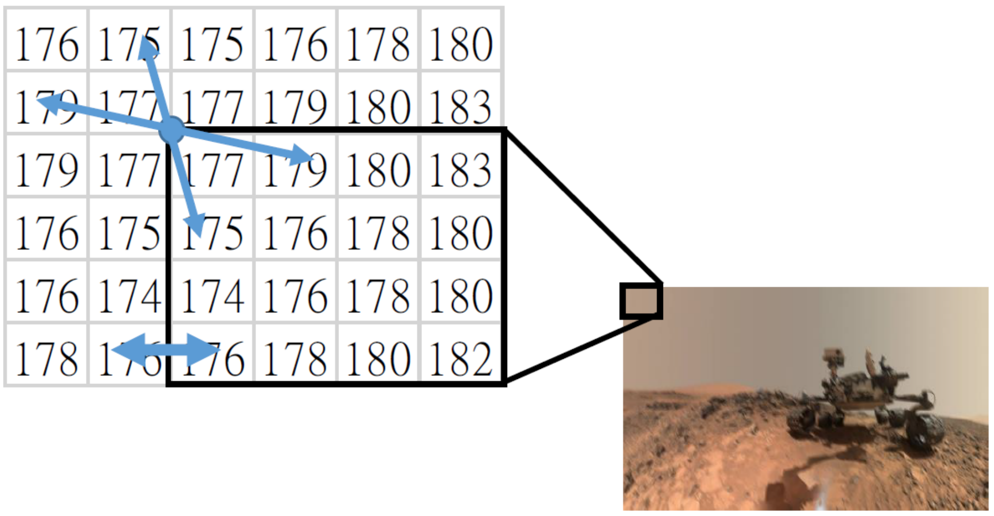
Put required files listed below in folder HW2_{studentID}

1. Your deblur_functions.py and TV_functions.py in folder /code
2. Your result deblurred images of problem 2.1 b in folder /MyDeblur_result
3. Your report with filename {studentID}_report.pdf

Compress your whole folder HW2_{studentID} in HW2_{studentID}.zip and submit to eeclass.

Note that wrong file delivery or arrangement will get 5% punishment.

4 Appendix



176	175	175	176	178	180
179	177	177	179	180	183
179	177	177	179	180	183
176	175	175	176	178	180
176	174	174	176	178	180
178	176	176	178	180	182

Figure 3: symmetric boarder condition

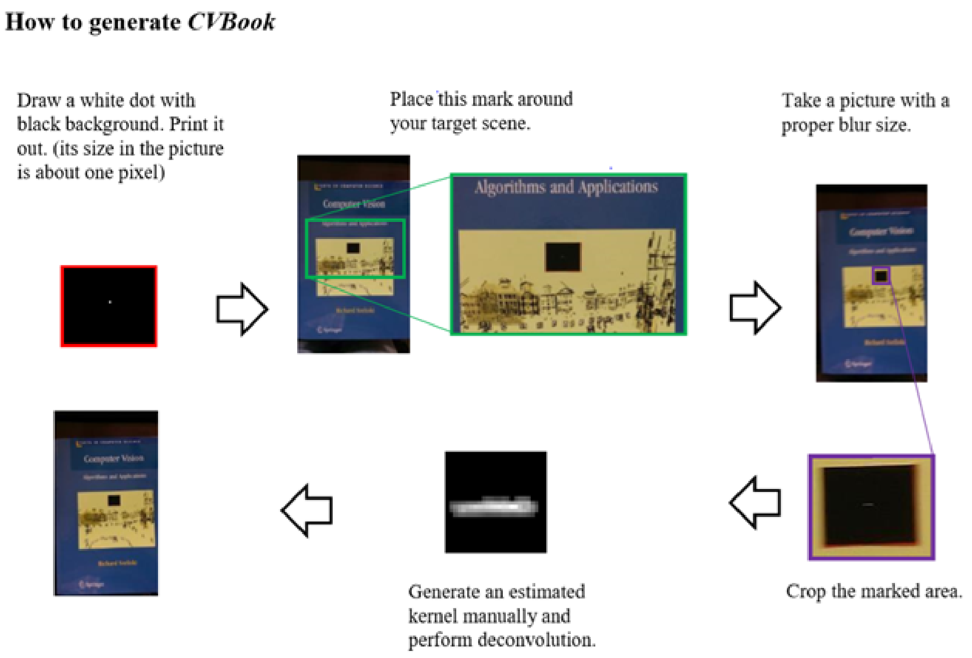

How to generate *CVBook*
Draw a white dot with black background. Print it out. (its size in the picture is about one pixel)
Place this mark around your target scene.
Algorithms and Applications
Take a picture with a proper blur size.
Crop the marked area.
Generate an estimated kernel manually and perform deconvolution.

Figure 4: How to generate CVBook

References

[1] W. H. Richardson, "Bayesian-based iterative method of image restoration," in *Journal of the optical society of America*, vol. 62, pp. 55–59, 1972.

[2] L. B. Rucy, "An iterative technique for the rectification of observed distributions," vol. 79, pp. 745–754, 1974.

[3] L. L. Yuan, J.Sum and H.-Y. Shum, "Progressive inter-scale and intra-scale non-blind image deconvolution," vol. 27, pp. 1–10, 2008.

[4] F. Heide, S. Diamond, M. Nießner, J. Ragan-Kelley, W. Heidrich, and G. Wetzstein, "Proximal: Efficient image optimization using proximal algorithms," vol. 35, pp. 1–15, 2016.