Computational Photography

Programming Assignment 2

Deblurring

Sagnik Ghosh

March 2022

1 Deblurring with Conventional Camera

In this task, we are given the clean image and we are given the exposure time of 52 seconds. In the real world, the pixel values are an integration of the light captured. For simplicity, we discretize the problem by capturing translated images at intervals of 1 second for a total of 52 seconds.

1.1 Generating Blurred Image

Figure 1: Blurred Image

1.2 Generating Blur Matrix

Note that the displayed Blur Matrix is for **vertical** blur, but the given image has **horizontal** blur. Consequently, we need to apply this blur kernel to the **transpose** of the image (as has been done in the implementation).

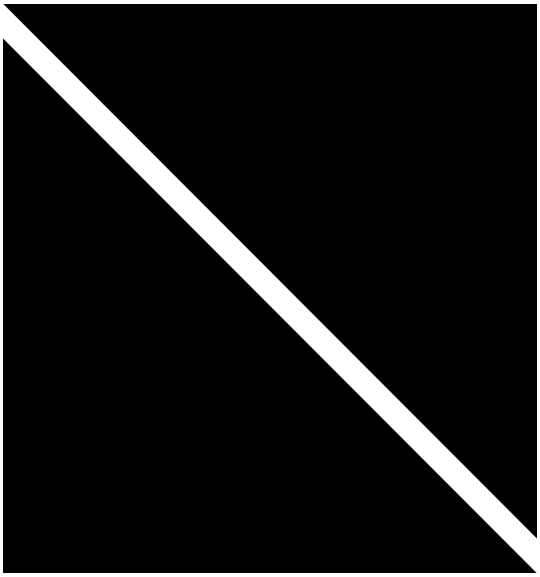

Figure 2: Blur Matrix

1.3 Deblurring

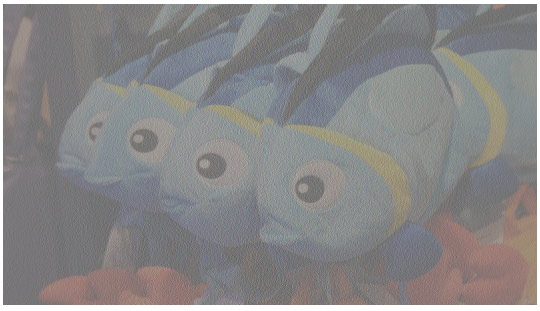

Figure 3: Deblurred Image

The **RMSE** between the clean image and the deblurred image is 0.3339.

1.4 Observations

- We can clearly notice that the deblurred image is very noisy as compared to the given clean image.
- The above observation can be explained as follows. The blur kernel for motion blur using a Conventional Camera is a box filter. The DFT of a box filter is the sinc function which has multiple zeroes. For deblurring (deconvolution), we divide the blurred image (which also contains noise) by the blur filter in the frequency domain. As the blur filter has many zeros, dividing by it amplifies the noise.

2 Deblurring with Flutter Shutter

In this task, we are given 2 binary exposure codes and the shutter remains open during the exposure time, only when the code is 1. This gives rise to blur kernels different from box filter (as in Conventional Camera). The 2 exposure codes we are given are:

- Code 1 - 1010000111000001010000110011110111010111001001100111
- Code 2 - 1010101010101010101010101010101010101010101010101010

2.1 Generating Blurred Images

Figure 4: Blurred Image using Code 1

Figure 5: Blurred Image using Code 2

2.2 Generating Blur Matrices

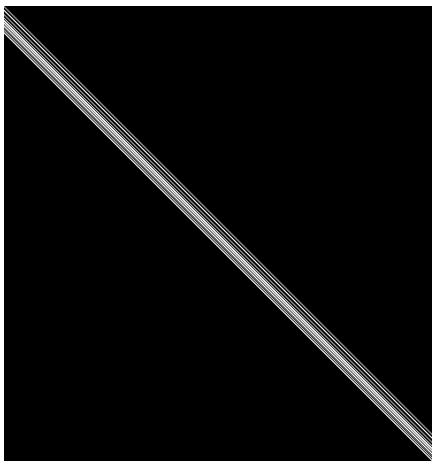

Figure 6: Blur Matrix using Code 1

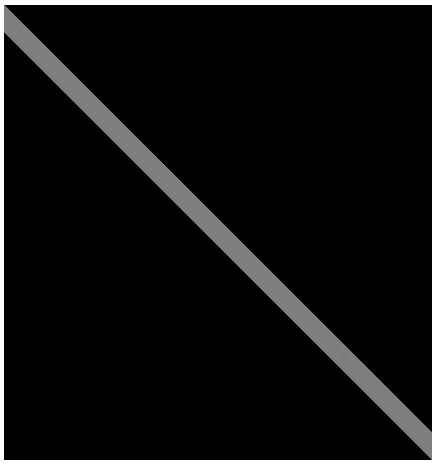

Figure 7: Blur Matrix using Code 2

2.3 DFT Plots

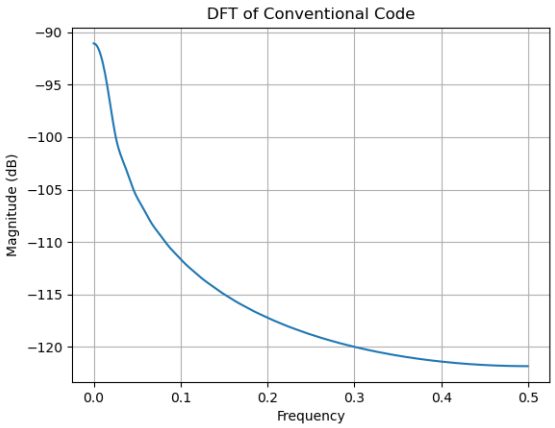

Figure 8: DFT of Blur Kernel - Conventional Camera

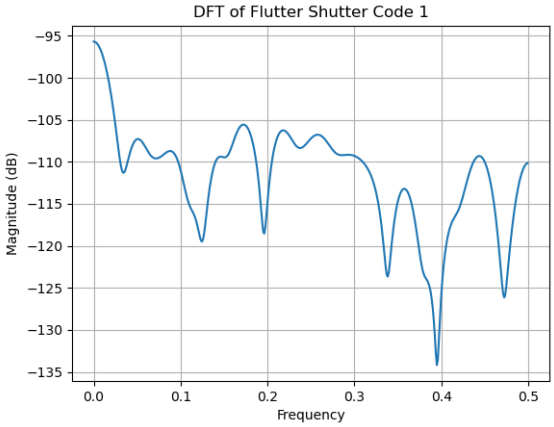

Figure 9: DFT of Blur Kernel - Code 1

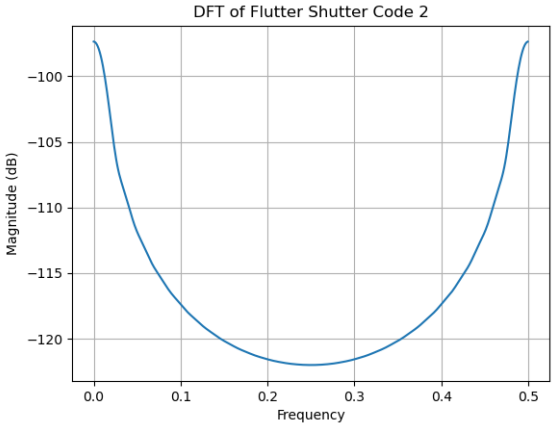

Figure 10: DFT of Blur Kernel - Code 2

- From the DFT plots, we can observe that the DFT of the blur kernel of the Conventional Camera has high values in the low-frequency region and it falls off near the high-frequency region. As the high frequency information is lost (low-pass filter), this is not optimal for recovering the deblurred image.
- The DFT plot of the blur kernel generated by using Code - 1 does not have many zeroes in either the low or high frequency regions. Hence, both low and high frequency information are preserved and it can be used to recover the deblurred image reliably. This reinforces our prior knowledge about this code as this is the optimal code arrived at by the authors of the Flutter Shutter paper, and they claim that the DFT of this code has minimum variance and does not have many zeroes.
- The DFT plot of this code is near zero for a considerable interval. We can notice that is code has alternate 0s and 1s, which is a high-pass filter. Due to zero-padding, the DFT plot is not exactly a high-pass filter, because it has a peak near zero frequency. Nonetheless, information is lost over a significant frequency range, rendering this code suboptimal for deblurring.

2.4 Deblurring

Figure 11: Deblurred Image using Code - 1

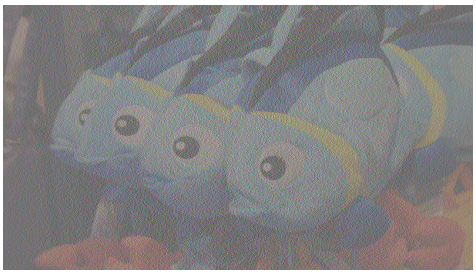

Figure 12: Deblurred Image using Code - 2

2.5 RMSE Values

- The **RMSE** between the clean image and the deblurred image using Code - 1 is 0.0652.
- The **RMSE** between the clean image and the deblurred image using Code - 2 is 0.3336.
- The RMSE values can be explained by considering the DFT plots above. The RMSE value of the deblurred image using Code - 1 is significantly lower than that of Code - 2. This is because the DFT of the blur kernel of Code - 2 is zero for a large frequency range, and during deconvolution, dividing by this kernel in the frequency domain amplifies the noise considerably. Whereas, the DFT of the blur kernel using Code - 1, is non-zero over almost all frequencies. Hence, the deblurred image using Code - 1 is less noisy and has a lower RMSE.

2.6 Deblurring without Noise

Figure 13: Deblurred Image using Conventional Camera without Noise

Figure 14: Deblurred Image using Code - 1 without Noise

Figure 15: Deblurred Image using Code - 2 without Noise

We can notice that the deblurred outputs for all 3 cases are not very noisy. If we conpute the RMSE between these deblurred images and the given clear image, it is of the order of 10^{-10}, which is near perfect reconstruction. This is because there is no additive noise on the blurred image. Hence, during deconvolution, while dividing the blurred image by the kernel in the frequency domain, the noise cannot get amplified due to zeroes, because there is no noise. The number of zeroes in the DFT of the blur kernel is inconsequential. Hence, all 3 blur kernels give near perfect reconstructions.

3 Motion Invariant Photography

In this task, the main idea is to make the blur kernel for objects with different velocities in the scene approximately the same, so that the same blur kernel can be used for the whole image for deblurring, eliminating the need for segmentation. This can be done by translating the camera in a parabolic path with respect to time, modifying the integration curve into a parabolic trajectory which is invariant to shear.

3.1 Generating Blurred Image with Static Camera

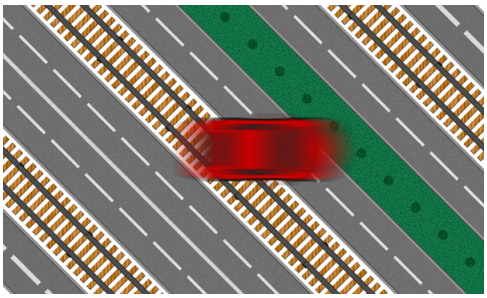

Figure 16: Blurred Image with Static Camera

3.2 Motion Invariant Blurred Image

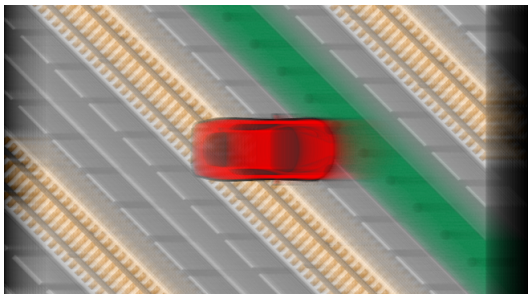

Figure 17: Motion Invariant Blurred Image

3.3 Observations

- The main observation we can draw is that in the output of the static camera, only the car is blurred because the background is static with respect to the camera. Whereas, in the second output, both, the car and the background are blurred. This is because the camera follows a parabolic path with respect to time, both the car and the background translate with respect to the camera.
- Another observation we can make is that the blur in the second image is approximately equal throughout the image, regardless of velocity. This is the main idea of Motion Invariant Photography, wherein the camera follows a parabola with respect to time. As a parabola is shear invariant, the PSFs of objects of all velocities are approximately the same.

3.4 PSF Plot

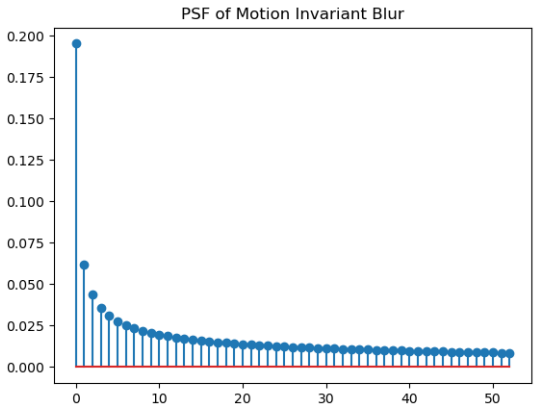

Figure 18: Motion Invariant Blur PSF Stem Plot

3.5 Generating the Blur Matrix

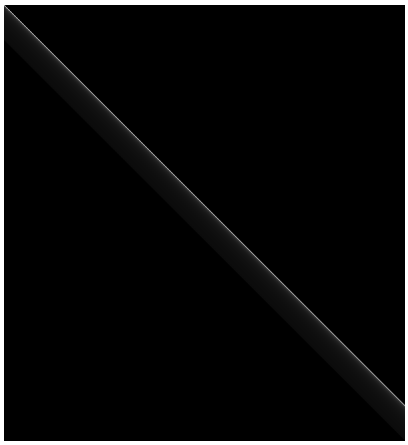

Figure 19: Motion Invariant Blur Matrix

3.6 Deblurring

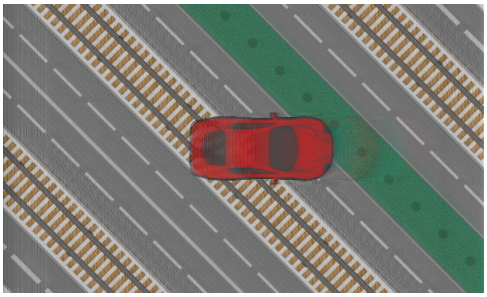

Figure 20: Motion Invariant Deblurred Image