Histogram Equalization

Li Kaiyu (S220048)
Chen Yueqi (G2202490B)
Chang Lo-Wei (G2203324E)

Assignment 01
Computer Vision
School of Computer Science and Engineering
Nanyang Technological University

2022

Table of Contents

Chapter 1

HE Algorithm Implementation

The main idea of histogram equalization (HE) is to flatten the histogram of intensity by a mapping function *f(I)*:

$$f(I) = (L-1)\sum_{j=o}^{I} p_j \tag{1.1}$$

Note that the image is processed in RGB format, and we will discuss more on selecting which color model to adopt in Chapter 3. To better indicate the process of HE, we use *sample07.jpg* as an example.

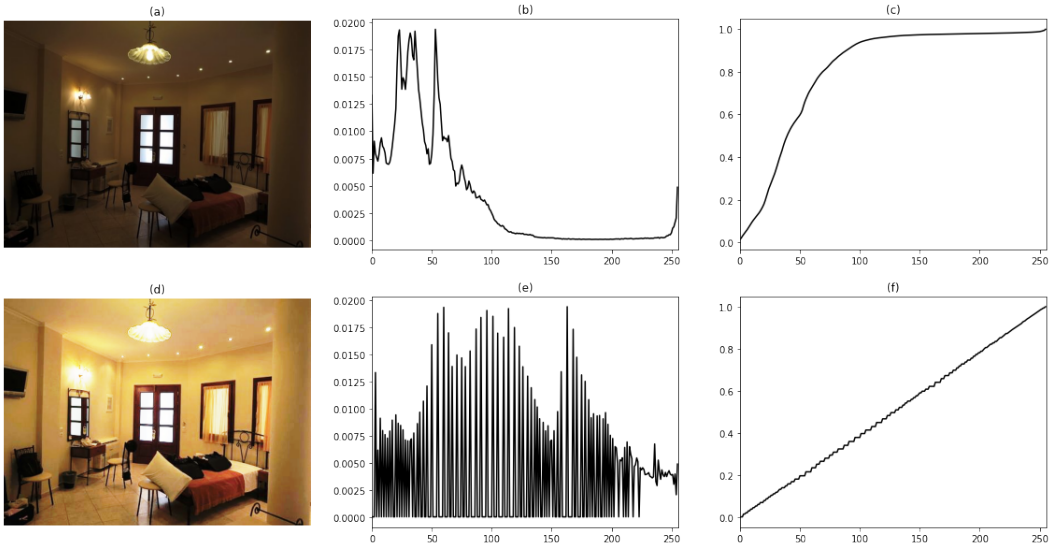

Figure 1.1: Image sample07: (a) original image (b) probability mass function (c) cumulative distribution function (d) histogram equalization image (e) HE PMF (f) HE CDF

1.1 Calculate the Histogram (Probability Mass Function)

First of all, we require the histogram of the image, which can be represented as probability distributions or simply bin counts. Here, we choose the probability mass function

(PMF) to obtain p_j in (1.1):

```
img = cv2.imread(imgStr)#image read by OpenCV
hist, bins = np.histogram(img.flatten(), 256, [0, 256])#bin counts
pmf = hist/np.sum(hist)#pmf
```

Note that the inputs are color images resulting in a three-dimensional matrix obtaining by *cv2.imread()*. Thus, we need to use *flatten()* to convert it to 1D before constructing the histogram.

The histogram *hist* we get contains the frequency of each intensity value. To calculate the PMF, we need to divide it by the sum using *np.sum()*. The result PMF is shown in Figure 1.1b.

1.2 Calculate the Cumulative Distribution Function

Secondly, cumulative distribution function (CDF) is required to compute $\sum_{j=o}^{I} p_j$ in (1.1). By simply applying *np.cumsum()*, the result CDF is shown in Figure 1.1c.

```
cdf = np.cumsum(pmf)#cdf
```

1.3 Calculate the Intensity Mapping Function

Recall the intensity mapping function *f(I)*:

$$f(I) = (L-1)\sum_{j=o}^{I} p_j$$

where L=256 since we have 256 bins from [0, 255]. We are now able to calculate $f(I)$ by replacing $\sum_{j=o}^{I} p_j$ with *cdf*.

```
mapping = np.round((256-1)*cdf).astype("uint8")#f(I) or s_k
```

Note that we add *np.round()* in the mapping since we need to round f(I) to an integer.

1.4 Apply Transformation to the Original Image

At last, we apply the intensity mapping function to the original image and get the result of histogram equalization.

```
HE = np.array([mapping[i] for i in img])
```

To understand the transformation method, we first need to know that *mapping* is an array containing $f(0)$ to $f(255)$. When considering its index as I, it becomes a dictionary in which the output value is $f(I)$ corresponding to its input key I.

The HE enhances the contrast of the original image, as shown in Figure 1.1d. Also, according to the PMF and CDF of HE in Figure 1.1e & Figure 1.1f, the histogram is flatter compared with the original histogram.

Chapter 2

Pros and Cons

2.1 Pros

As we illustrated in Chapter 1, HE appears to be a simple and effective method to improve the contrast of image. This method provided a solid backbone and inspired many powerful techniques to be developed based on it. For instance, the variations of HE includes Bi-Histogram Equalization(BBHE)[1], Quadrant Dynamic Histogram Equalization(QDHE)[4], Contrast Limited Adaptive Histogram Equalization(CLAHE)[5] and so on. In the next chapter, we will demonstrate how to address the problems of HE by various modifications on the simple HE approach.

2.2 Cons

2.2.1 Visible Image Gradient

As shown in Figure 2.1, after performing HE on *sample01.jpg*, some unexpected chroma noises appear in the resulting image and the result of *sample02.jpeg* is not satisfactory since irregular color blocks almost fill the entire image.

This is mainly because the image gradient become visible. By definition, an image gradient is a directional change in the intensity or color in an image[6]. Known that two adjacent pixels in a flat region should have a very small and invisible gradient. However, after HE, two adjacent pixels of the similar color will become different colors, where the gradient becomes visible. Because HE will shift the high frequency intensities to the lower ones;, but three channels (RGB) won't have the same shifting distance.

Figure 2.1: Image sample01/02: original vs HE

2.2.2 Overexposure/Underexposure

Figure 2.2 shows that *sample08.jpg* is overexposed after performing HE while *sample03.jpg* is underexposed. An underexposed image is the sort of photograph that one might consider to be too dark while an overexposed image is considered to be too bright.

By comparing before-and-after histograms of these images, we notice that underexposure happens when the original image is too bright so that its histogram will concentrate on the high intensity side (the brighter side), as shown in Figure 2.3(left). To have an evenly distributed histogram, the frequency of high intensity side should be reduced to increase that of low intensity side and the shifting result is shown in Figure2.3(right). Consequently, after HE, the lower intensity side will have unexpectedly high frequency, resulting in an underexposed area as shown in Figure2.2. Vice versa for overexposure.

Figure 2.2: From top left to bottom right: (1)original sample03, (2)sample03 underexposure area, (3)original sample08, (4)sample08 overexposure area

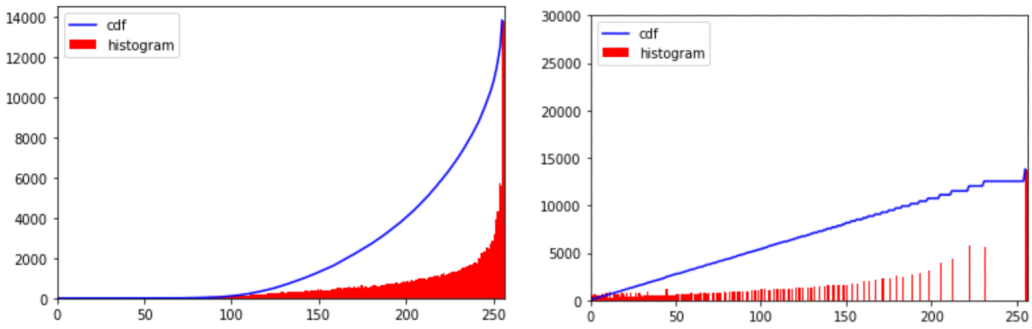

Figure 2.3: Image sample03: Before-and-After Histogram

2.2.3 Increase Contrast of Background Noise

By looking at the upper-left corner of improved *sample01.jpg*, we notice that there are some noises appearing while the original image does not have as shown in Figure2.4. In other words, current HE algorithm will enhance the contrast of noise as well.

It's worth noting that these noises all appear on the white background area in the image. Different from other 7 sample images, *sample01.jpg* is the only one that has a large white background. Since current HE algorithm evenly distributed the intensity without distinguishing background and foreground, the intensity of pixels in the white background will be effected by the foreground.

2.2.4 Blocking Artifact

Figure 2.5 shows that the output images of HE algorithm suffer from a lot of blocking artifacts. However, if we compare the output images with the original images, we

Figure 2.4: Image sample01: original vs HE

will discover that those blocking artifacts already exists in original images, and they become more visible after applied HE. Notice that all the given input images are in JPEG format, and this kind format adopts the discrete cosine transform algorithm for compression. This a lossy compression that generates blocking artifacts[2].

Figure 2.5: Image sample06/08: original vs HE

2.2.5 Invertiblity and Information Loss

HE is not an invertible approach, that means if given an output equalized image as well as the mapping function, we are not able to recover the original input image. Recall figure 1.1(c), as we are using the rescaled and rounded CDF as the mapping function, any horizontal part within the function will map different input intensity values to a single output value. When this happen, some information is lost, and the mapping function will be no longer invertible.

Chapter 3

Improvements

3.1 HSV model vs RGB model

Figure 3.1 shows the results when we apply HE on different image color models. From left side to right side, the first image is the original image. We merged the RGB channel together to calculate a single HE mapping function and acquired the second image. The third image is obtained when we treat RGB channel separately and thus calculate three different mapping functions for each channel. The latter approach on the given image set generally exhibits visible color distortion so the former way is adopted when we use RGB model.

We also tried HE on HSV color model instead(the forth image in figure 3.1), where H is the hue component, S is the saturation component, and V is the value component. Due to the defination and a hue discontinuity around 360°, HE can only be applied on V and S components[7]. The equalized HSV image appears to have a more sensible color overall compared to RGB result but the color tends to be over-saturated. In this report, our improvements mainly focus on the HE algorithm itself rather than a better color model for applying HE, so we leave the choice of color model as a trade-off.

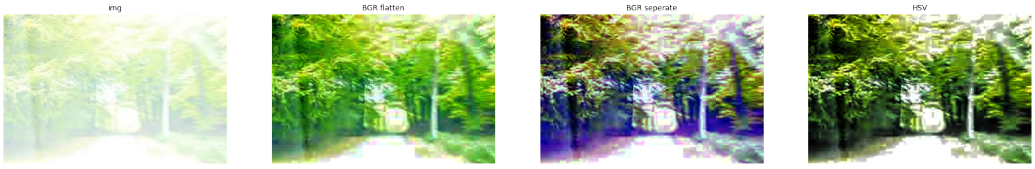

Figure 3.1: Image sample04: RGB vs HSV

3.2 Partial Histogram Equalization

As described in chapter 2.2.2, HE algorithm evenly equalize the original color intensity to the full space of intensity value(0 - 255), and this may result in severe overexposure or under exposure. To compensate for this side effect, a straightforward way is to only partially adopt HE result. Therefore, we can define a new mapping function $f_{partial}(I)$

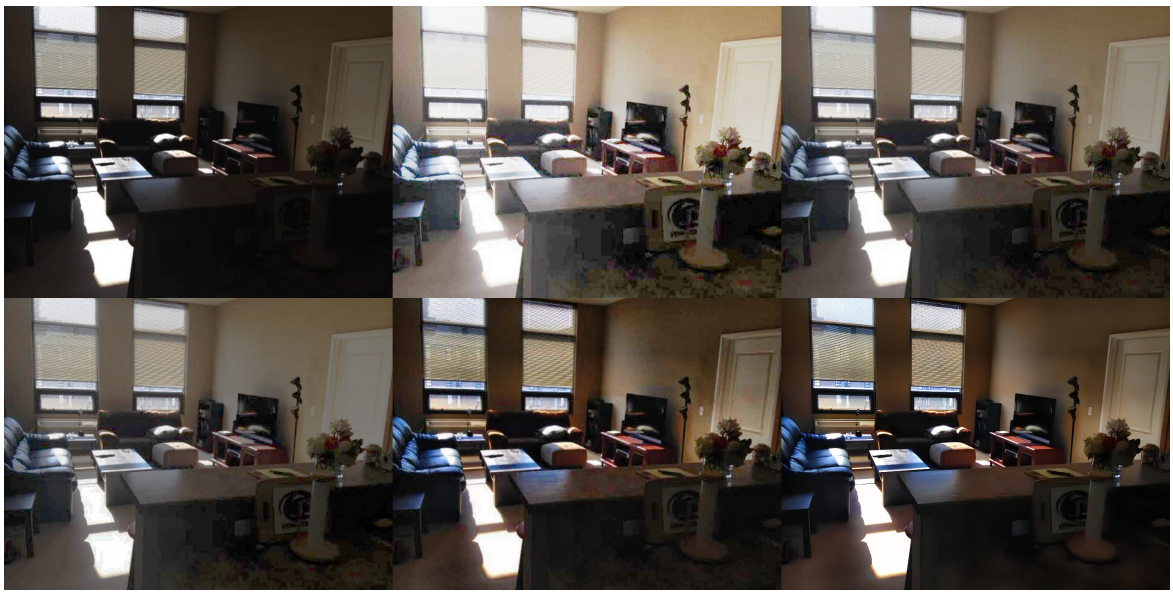

Figure 3.2: From top left to bottom right: (1)original input, (2)basic HE, (3)partial HE, (4)BBHE, (5)CLAHE, (6)CLAHE on preprocessed image

based on function $f(I)$(1.1):

$$f_{partial}(I) = \alpha f(I) + (1-\alpha)I \tag{3.1}$$

This is the partial HE function, which defines a linear blend between an identity function and the HE mapping function(the rescaled cumulative distribution function), controlled the by a coefficient α. Result in figure 3.2 illustrates how partial HE mitigates the overexposure defect, as well as maintain original color distribution.

3.3 Bi-Histogram Equalization

Considering the overexposure/underexposure problem mentioned in chapter 2.2.3, one solution is to limit the shifting range when we shift the high frequency intensities to the lower ones. With this motivation, we can modify original HE to Bi-histogram equalization (BBHE). That is:

Suppose X_m is the average intensity of image X and X_i is the intensity value where

$$X_m \in \{X_0, X_1, ..., X_{255}\}$$

Based on X_m, we separate the intensity into X_L and X_U such that

$$X = X_L \cup X_U$$

Then we perform HE on X_L and X_U respectively and merge the output to get the final results. Inspired by Bi-HE, we implemented n-HE, which is to separate the intensity into n parts and perform HE on each part respectively.

Compared Figure3.2(2) with Figure3.2(4), BBHE solves the overexposure problem in the table area in *sample08.jpg*.

3.4 CLAHE

The basic HE algorithm applies globally on the entire image, which leads to many problems such as over amplification(see chapter 2.2.2) and undesired image contrast enhancement on backgrounds noise(see chapter 2.2.3). A natural solution is to divide the image into several blocks and then perform basic HE on each block. However, this simple solution exhibits strong blocking artifacts, due to the discontinuities of intensity between blocks[6]. This can be addressed by adopting an interpolation function within blocks, which is called as adaptive histogram equalization(AHE).

Figure 3.2(5) shows the result of adopting CLAHE[5], which is a gain-limited version of AHE, and in general this method achieves the best results in our project.

3.5 Image Preprocessing with SwinIR

Figure 3.2(5) and Figure 3.3 show the CLAHE(see section 3.4) result of image sample06 and sample, that still exhibit detectable block artifacts. These blocks are not caused by blocks used in the CLAHE algorithm, but already exists in the original JPG image(see section 2.2.4), then amplified by the HE process. To remove these artifacts, the input image needs first to be preprocessed. Here we directly apply a powerful model known as SwinIR[3] to handle this work. SwinIR is a deep learning based model that achieves state-of-the-art performance on image denosing, JPEG compression artifact reduction and image Super Resolution. In figure 3.3, we can clearly observate that the block artifacts within both red circles are completely removed by SwinIR.

Figure 3.3: Image sample08: CLAHE without preprocessing vs SwinIR + CLAHE

Bibliography

[1] Yeong-Taeg Kim. Contrast enhancement using brightness preserving bi-histogram equalization. *IEEE Transactions on Consumer Electronics*, 43(1):1–8, 1997.

[2] Weihai Li, Yuan Yuan, and Nenghai Yu. Passive detection of doctored jpeg image via block artifact grid extraction. *Signal Processing*, 89(9):1821–1829, 2009.

[3] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 1833–1844, October 2021.

[4] Chen Hee Ooi and Nor Ashidi Mat Isa. Quadrants dynamic histogram equalization for contrast enhancement. *IEEE Transactions on Consumer Electronics*, 56(4):2552–2559, 2010.

[5] Stephen M. Pizer, E. Philip Amburn, John D. Austin, Robert Cromartie, Ari Geselowitz, Trey Greer, Bart ter Haar Romeny, John B. Zimmerman, and Karel Zuiderveld. Adaptive histogram equalization and its variations. *Computer Vision, Graphics, and Image Processing*, 39(3):355–368, 1987.

[6] Richard Szeliski. *Computer Vision Algorithms and Applications*. Springer, 1st edition, 2011.

[7] Wan Yussof. Performing contrast limited adaptive histogram equalization technique on combined color models for underwater image enhancement. 01 2013.