**Computer Vision HW3: Histogram Equalization**

**R10741015 鄭傑鴻**

**Sept. 20, 2022**

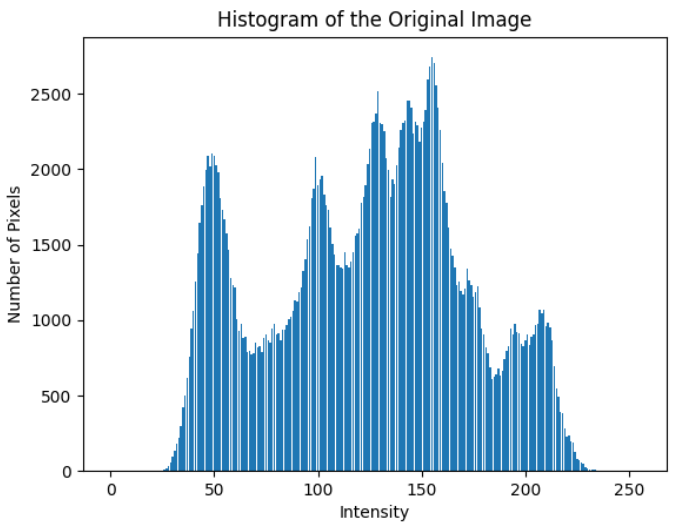
(a) Original image and its histogram

- Description:
  1. Initiate a new array $count\_array$ with shape (256, 1), where $count\_array[i]$ stores the number of pixels with intensity *i*.
  2. Iterate over the original picture and accumulate numbers for different intensities in $count\_array$.
  3. Visualize the result using matplotlib
- Code:

```
def generateHistogram(self, img, filename, graphname, ret=False):
    def plotHistogram(ys, title='', save=True):
        plt.bar(range(256), ys)
        plt.title(graphname)
        plt.xlabel('Intensity')
        plt.ylabel('Number of Pixels')
        plt.savefig('histogram_{}.png'.format(filename))
        plt.clf()

    count_array = np.zeros((256,), dtype=np.uint32)
    for i in range(512):
        for j in range(512):
            count_array[img[i][j]] += 1
    plotHistogram(count_array, title=filename)
    if ret:
        return count_array
```

- Resulting Image:

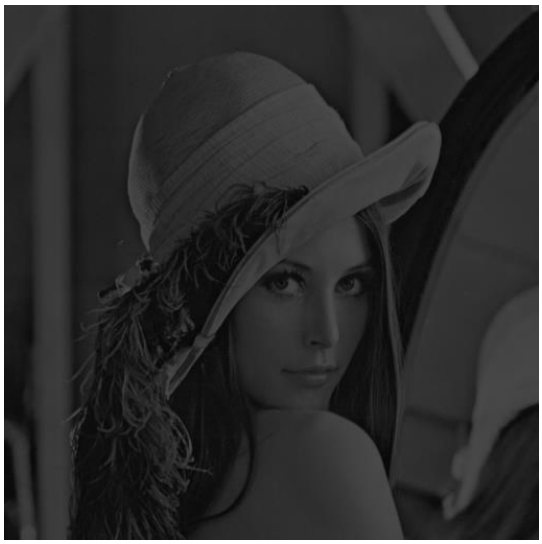


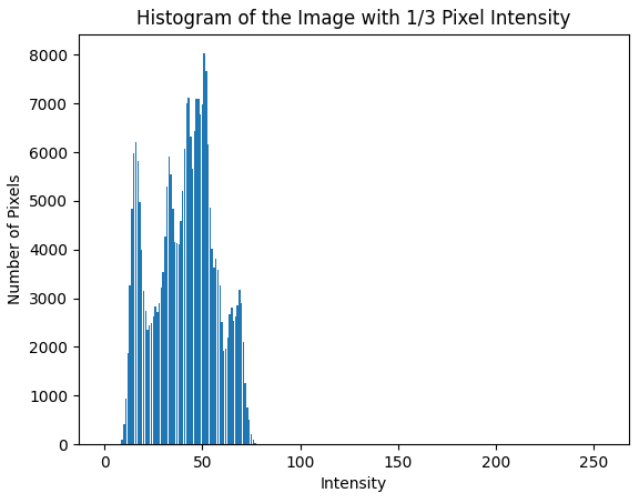
(b) Image with intensity divided by 3 and its histogram

- Description
  1. Initiate a new zero array $pic\_new$, with the same shape as the original picture (512, 512), to store the new picture.
  2. Iterate over the original picture $pic$, assign pixel intensity divided by three, i.e. $int(pic[i][j]/3)$, to $pic\_new[i][j]$($i$ and $j$ both from 0 to 511)
  3. Show the image with 1/3 intensity.
  4. Reuse the code from part (a) above to generate histogram, and return an $count\_array$ filled with numbers for each pixel intensity.
- Code

```
def divideByThree(self):
    pic_new = np.zeros(self.pic.shape, np.uint8)
    for i in range(512):
        for j in range(512):
            pic_new[i][j] = int(self.pic[i][j]/3)
    return pic_new
```

- Result





(c) Image after applying histogram equalization to (b) and its histogram

- Description

  Consider the relationship:

$$s_k(new\ intensity\ for\ original\ intensity\ k) = 255 \times \sum_{j=0}^{k} \frac{n_j(\#\ of\ pixels\ with\ intensity\ j)}{n(total\ number\ of\ pixels = 512^2)}$$

  1. Set up a new zero array $s\_array$, with shape (256, 1), to store information about $s_k, k\ from\ 0\ to\ 255$
  2. Utilize the $count\_array$ from part (b). With index *k* from 0 to 255, calculate the corresponding $s_k$ by the formula above.
  3. Initiate a new zero array $pic\_new$, with shape (512, 512), to store the new picture.
  4. Iterate over the picture with 1/3 intensity, and translate each pixel intensity $pic\_div[i][j]$ to the corresponding new intensity using the $s\_array$
  5. Show the picture with equalized intensity
  6. Reuse code from part (a) to generate histogram

- Code

```
def equalization(self, pic_div3, histo_array):
    s_array = np.zeros((256,), dtype=float)

    cumulate = 0
    for k in range(256):
        cumulate += 255 * histo_array[k] / (512**2)
        s_array[k] = cumulate

    pic_new = np.zeros(self.pic.shape, np.uint8)
    for i in range(512):
        for j in range(512):
            pic_new[i][j] = int(s_array[pic_div3[i][j]])
    return pic_new
```

- Result