**Computer Vision HW3: Histogram Equalization**

**R10741015 鄭傑鴻**

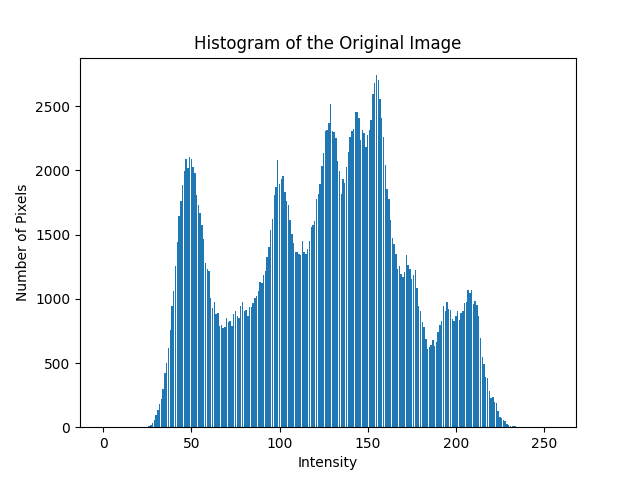
**Sept. 20, 2022**

1. Original image and its histogram

* Description:
  1. Initiate a new array with shape (256, 1), where stores the number of pixels with intensity *i*.
  2. Iterate over the original picture and accumulate numbers for different intensities in .
  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:

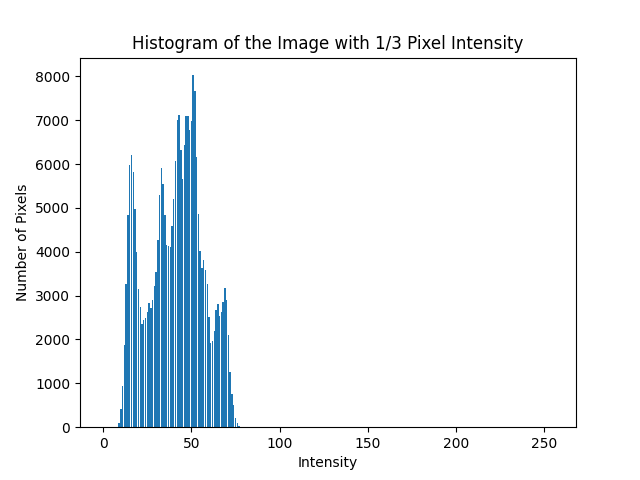
****

1. Image with intensity divided by 3 and its histogram

* Description
  1. Initiate a new zero array , with the same shape as the original picture (512, 512), to store the new picture.
  2. Iterate over the original picture , assign pixel intensity divided by three, i.e. , to (*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 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



1. Image after applying histogram equalization to (b) and its histogram

* Description

Consider the relationship:

* 1. Set up a new zero array , with shape (256, 1), to store information about
  2. Utilize the from part (b). With index *k* from 0 to 255, calculate the corresponding by the formula above.
  3. Initiate a new zero array , with shape (512, 512), to store the new picture.
  4. Iterate over the picture with 1/3 intensity, and translate each pixel intensity to the corresponding new intensity using the
  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

