Image Processing: Assignment #1

Problem 1 – Sampling, Quantization

a) The decision on how many megapixels a camera has, or how big the resolution is of a computer-generated image should lead to balance between image quality, performance and practical consideration.
When we think about steaming service, we want to get a good quality video (high resolution frames), but we need to consider limitation of the internet performance so that the streaming will smooth. Another example of quality - performance balance is in autonomous vehicles that use cameras for their operation. The image processing should be performed at a very fast pace, so the camera should have low resolution video, but not too low to avoid missing part of the feature in the frame.
We also need to consider the image size according to the hardware limitation, for example some systems with low amount of memory. The image size should be small, meaning with a low number of pixels.
In addition, physical related reasons are also should be taken into consideration. One important example is the size of camera sensor. The larger the sensor's surface area, it can gather more light in a single shot. Larger camera sensors are excellent for low-light photography. Although more megapixels creating a higher-resolution image with more details, trying to fit a lot of megapixels on a smaller sensor creates problems when it comes to low-light photography.

b) Quantization is the second stage in the digitization process, in quantization stage, we choose number of gray levels according to number of assigned bits. The number of bits used to define each pixel is called BIT DEPTH. A lower bit depth means less possible color. Older computers had limited memory resources, so there should be a balance between the quality and the use of a memory for an image. Another consideration is the number of colors, old screens or other display gear can show, for example if a screen can display limited number of colors, there is no need for high bit depth (large number of bits to represent the pixel number)
To sum up, if we have a limited memory resources or less advanced screen, we need stronger quantization.

Problem 2 – Nyquist

a) $\sin(2\pi x)$ has a cycle in length 1. We saw in tutorial, for the sine wave $\sin(2\pi xk)$, for k>1 we shrink the wave, but for k<1 we stretch the wave. For $\sin(2\pi xk)\ with\ k = \frac{1}{2}$ we will get $\sin(\pi x)$ and its cycle in length 2 (stretched by a factor of 2). Thus $\sin(\pi xk)$ has wavelength in size $\frac{2}{k}$. Also, the value of A is not relevant because it defines the width of the sampled area in the sampling grid, and it's not relevant to the periodic image itself.

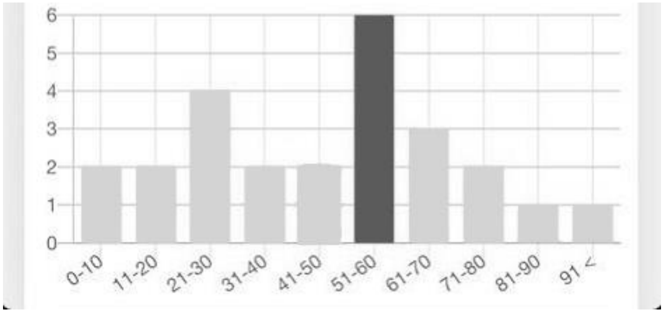
b) The wavelength of an image is $\frac{1}{frequency} \rightarrow frequency = \frac{1}{\text{wavelength}}$.
In this image $frequency_i = \frac{k}{2}$. According to Nyquist rule, to observe details at frequency f one must sample at frequency > 2f. Thus, the frequency of the sampling grid should be $frequency_{sgrid} > \frac{2k}{2} = k$. Therefore, k values should be lower than the sampling grid frequency.
Calculation of k for each value A = {0.25,2} and the distance unit is 1cm.

A=0.25: $frequency_{sgrid} = \frac{number\ of\ cycles}{distance\ unit} = \frac{2}{1} = 2 > k$

A=2: $frequency_{sgrid} = \frac{number\ of\ cycles}{distance\ unit} = \frac{\frac{1}{2}}{1} = \frac{1}{2} > k$

Problem 3 – Histograms, Matching, Quantization

e) By observing the lower part of the images in gray scale:



We need to distinguish between the bars and- grid+2 borders of the image (in same color), text and background. Thus, we choose the optimal number of gray levels to be 4 to distinguish between 4 ranges of grey.

Output

```
D:\venv\Image_Processing_labs\Scripts\pyth
Histogram a.jpg gave 2,2,4,2,2,6,3,2,1,1
Histogram b.jpg gave 6,2,1,1,3,3,6,2,2,3
Histogram c.jpg gave 0,0,0,0,0,0,1,1,1,1
Histogram d.jpg gave 1,0,2,3,4,3,5,5,6,2
Histogram e.jpg gave 2,1,1,3,2,5,1,1,2,3
Histogram f.jpg gave 1,0,1,1,1,4,1,1,2,1
Histogram g.jpg gave 1,1,1,3,1,2,9,3,3,0

Process finished with exit code 0
```