COMP 478/6771 Image Processing

Solutions to Assignment 1

Student: Qingbo Kang **Student ID: 40058122**

Part I: Theoretical questions

1. Nonlinear. An operator that computes the median of a set of pixels of a sub-image area which acts like media filter is nonlinear.
   The example is given below:

   Suppose $f(x,y)=\begin{bmatrix}1 & 4 & 100\\ 3 & 60 & 2\\ 7 & 9 & 4\end{bmatrix}$, $g(x,y)=\begin{bmatrix}6 & 8 & 0\\ 23 & 45 & 9\\ 85 & 34 & 13\end{bmatrix}$, $a=1, b=1$,

   Apply that operator, with symmetrically extended at the boundaries:

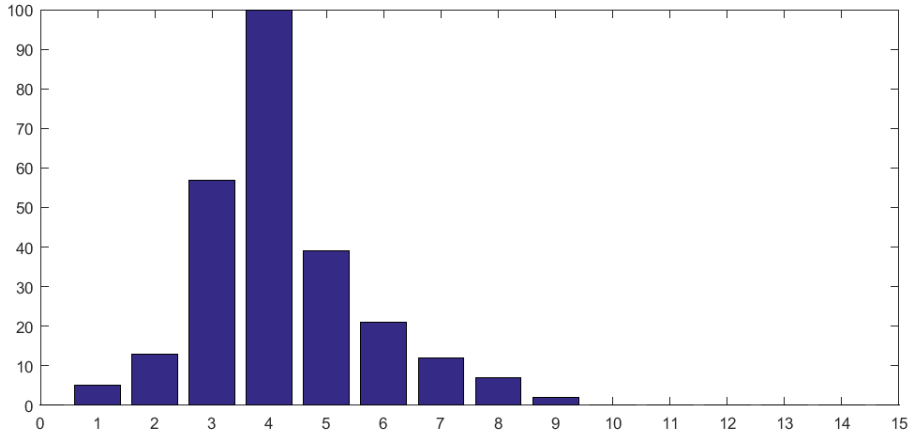
   $$H[f(x,y)]=\begin{bmatrix}3 & 4 & 60\\ 4 & 4 & 4\\ 7 & 7 & 4\end{bmatrix},\quad H[g(x,y)]=\begin{bmatrix}8 & 8 & 8\\ 23 & 13 & 9\\ 45 & 34 & 13\end{bmatrix},$$

   $$H[af(x,y)+bg(x,y)]=H\begin{bmatrix}7 & 12 & 100\\ 26 & 105 & 11\\ 92 & 43 & 17\end{bmatrix}=\begin{bmatrix}12 & 12 & 100\\ 26 & 26 & 17\\ 92 & 43 & 17\end{bmatrix},$$

   $$aH[f(x,y)]+bH[g(x,y)]=\begin{bmatrix}3 & 4 & 60\\ 4 & 4 & 4\\ 7 & 7 & 4\end{bmatrix}+\begin{bmatrix}8 & 8 & 8\\ 23 & 13 & 9\\ 45 & 34 & 13\end{bmatrix}=\begin{bmatrix}11 & 12 & 68\\ 27 & 17 & 13\\ 52 & 41 & 17\end{bmatrix}$$

   It clearly can be seen that:
   $H[af(x,y)+bg(x,y)]\neq aH[f(x,y)]+bH[g(x,y)]$
   So by this example, the operator that computes the median of a set of pixels of a sub-image area is nonlinear.

2. a) The histogram image shows below:



b)

i) First, from the table, the image is a 4-bit image, L = 16, image size = 256, intensity levels [0, 15].

Calculate r_k, n_k, $p_r(r_k)$:

r_k	n_k	$p_r(r_k)$	r_k	$p_r(r_k)$	$p_r(r_k)$
$r_0 = 0$	0	0	$r_8 = 8$	7	0.0273
$r_1 = 1$	5	0.0195	$r_9 = 9$	2	0.0078
$r_2 = 2$	13	0.0508	$r_{10} = 10$	0	0
$r_3 = 3$	57	0.2227	$r_{11} = 11$	0	0
$r_4 = 4$	100	0.3906	$r_{12} = 12$	0	0
$r_5 = 5$	39	0.1523	$r_{13} = 13$	0	0
$r_6 = 6$	21	0.0820	$r_{14} = 14$	0	0
$r_7 = 7$	12	0.0469	$r_{15} = 15$	0	0

Since $S_k = (L-1)\sum_{j=0}^{k}(\frac{n_j}{n}), k = 0,...,L-1$, therefore:

$S_0 = 15 \times 0 = 0$, $S_1 = 15 \times 0.0195 = 0.2925$, $S_2 = 15 \times (0.0195 + 0.0508) = 1.0545$,

$S_3 = 15 \times (0.0195 + 0.0508 + 0.2227) = 4.395$,

$S_4 = 15 \times (0.0195 + 0.0508 + 0.2227 + 0.3906) = 10.254$,

$S_5 = 15 \times (0.0195 + 0.0508 + 0.2227 + 0.3906 + 0.1523) = 12.5385$,

$S_6 = 15 \times (0.0195 + 0.0508 + 0.2227 + 0.3906 + 0.1523 + 0.082) = 13.7685$,

$S_7 = 15 \times (0.0195 + 0.0508 + 0.2227 + 0.3906 + 0.1523 + 0.082 + 0.0469) = 14.472$,

$S_8 = 15 \times (0.0195 + 0.0508 + 0.2227 + 0.3906 + 0.1523 + 0.082 + 0.0469 + 0.0273) = 14.8815$,

$S_9 = S_{10} = S_{11} = S_{12} = S_{13} = S_{14} = S_{15} = 15 \times 1 = 15$.

Because the gray levels are integers, so the values of S_k are:

$S_0 = 0$, $S_1 = 0$, $S_2 = 1$, $S_3 = 4$, $S_4 = 10$, $S_5 = 13$, $S_6 = 14$, $S_7 = 14$,

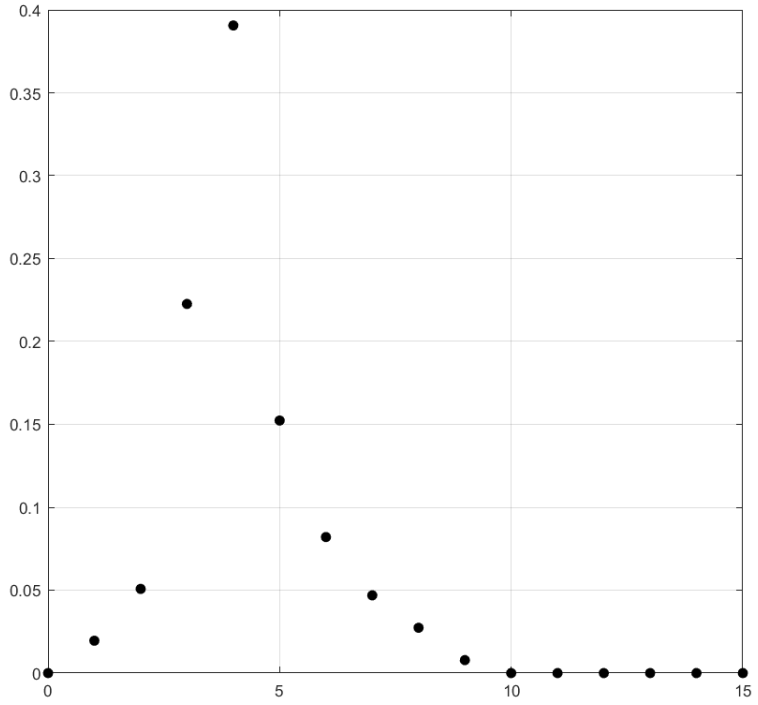
$S_8 = S_9 = S_{10} = S_{11} = S_{12} = S_{13} = S_{14} = S_{15} = 15$.

ii) First, computing $p_s(s_k)$:

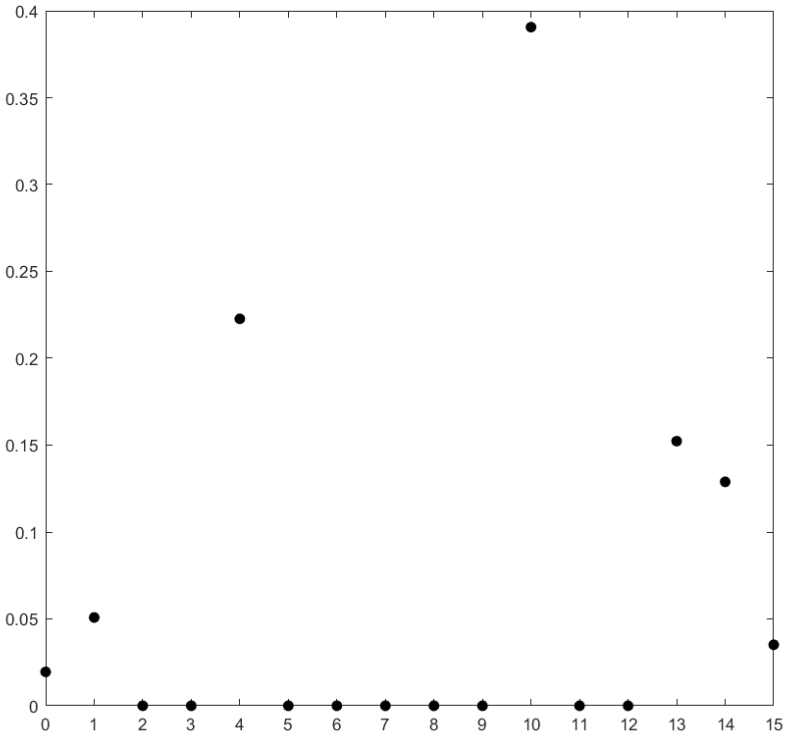
$p_s(s_0) = 0.0195$, $p_s(s_2) = 0.0508$, $p_s(s_3) = 0.2227$, $p_s(s_4) = 0.3906$, $p_s(s_5) = 0.1523$,

$p_s(s_6)$=0.1289, $p_s(s_8) = 0.0351$.

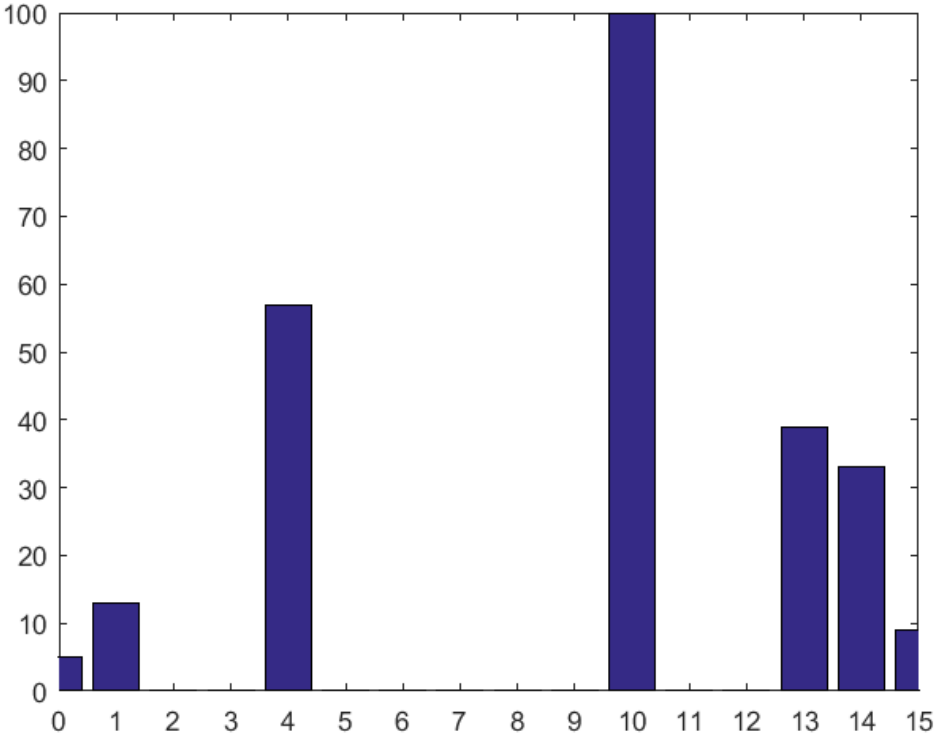
The below shows the figure of $p_r(r_k)$:



The below shows the figure of $p_s(s_k)$:



c) The below shows the new histogram after performing the histogram equalization:



d) Because the original histogram have a peak level which means this gray level have a relatively large amount of pixels (in this example, the gray level is 4, nearly 40% pixels are at this gray level), and the histogram equalization technique just mapping one gray level to another one, it's a is single valued and monotonically increasing transformation.
e) The same. A second pass of histogram equalization (on the histogram-equalized image) produce the same result as the first pass.

Explanation: Suppose n is the total number of pixels and n_{rj} is the number of pixels in the image with intensity value rj. According to the histogram equalization transformation:

$$s_k = T(r_k) = \sum_{j=0}^{k} \frac{n_{rj}}{n} = \frac{1}{n}\sum_{j=0}^{k} n_{rj}$$

Since every pixel with value r_k is mapped to value s_k. Then we have,

$$n_{sk} = n_{rk}$$

A second pass of histogram equalization:

$$u_k = T(s_k) = \frac{1}{n}\sum_{j=0}^{k} n_{sj}$$

and $n_{sj} = n_{rj}$, so we get:

$$u_k = \frac{1}{n}\sum_{j=0}^{k} n_{sj} = \frac{1}{n}\sum_{j=0}^{k} n_{rj} = s_k$$

And it means a second pass of histogram equalization produce the same result as the first pass.

3. First, apply the histogram equalization:

$$u = T(r) = \int_0^r p_r(k)dk = \int_0^r (-2k+2)dk = -r^2 + 2r$$

And for the second diagram:

$$v = H(z) = \int_0^z p_z(k)dk = \int_0^z 2kdk = z^2,$$

Let $u = v$, then

$$-r^2 + 2r = z^2$$

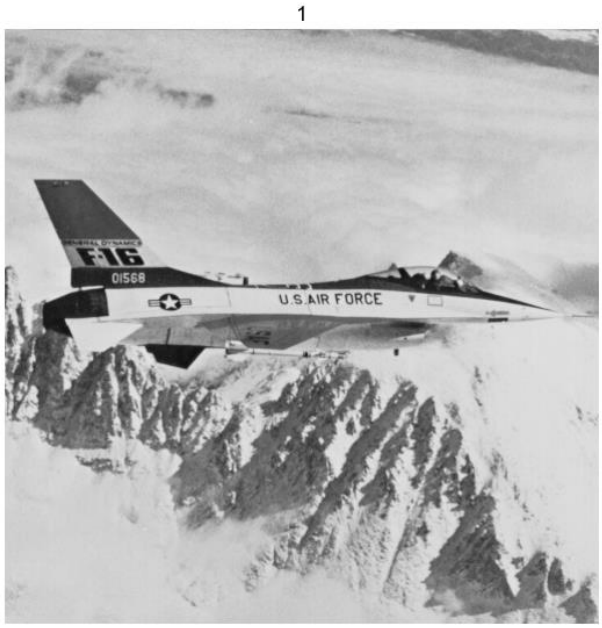
$$\Rightarrow z = \pm\sqrt{-r^2 + 2r}$$

Finally, because the intensity levels are all positive numbers, so

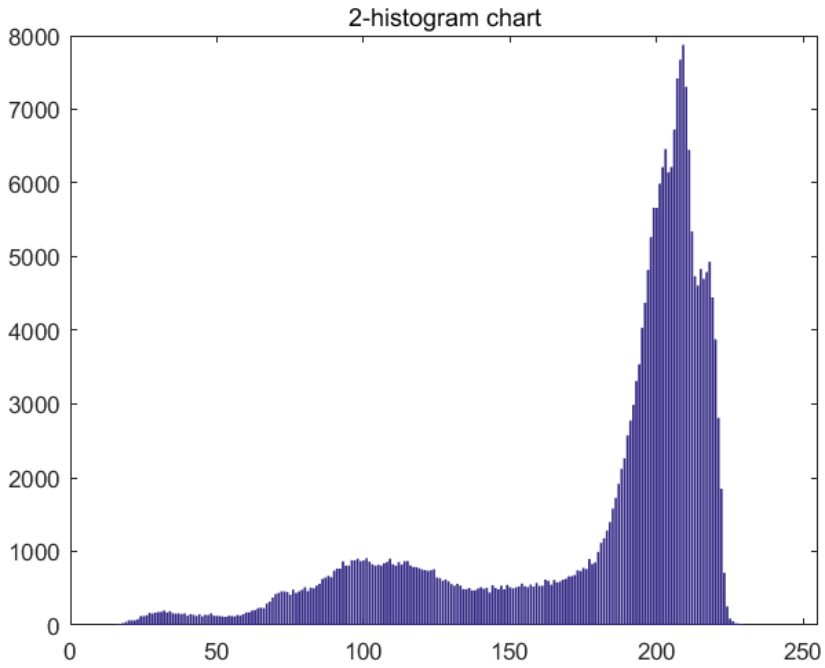
$$z = \sqrt{-r^2 + 2r}$$

Part II: Programming question

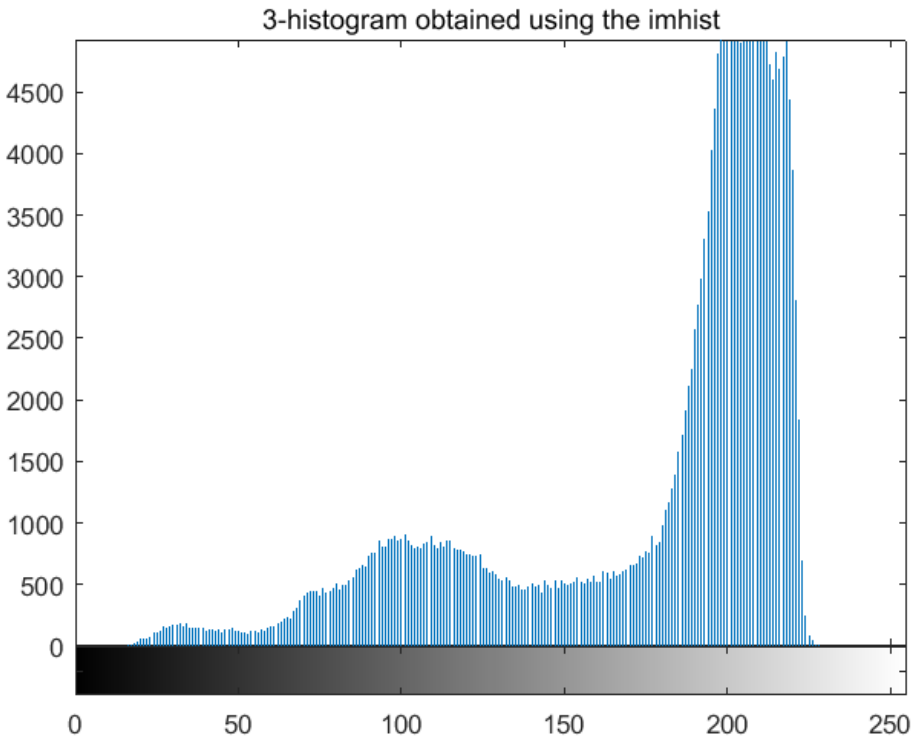
1) See code file Solution_1.m.
   Below shows the running figure:



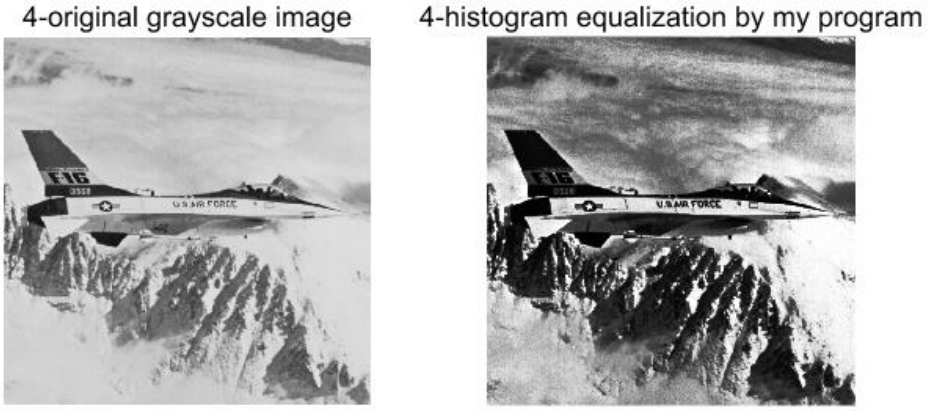
2) See code file Solution_1.m.
   Below shows the running figure:



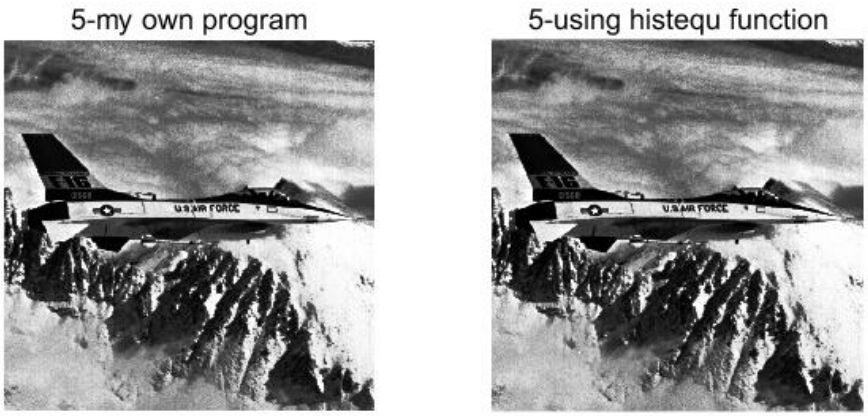
3) See code file Solution_1.m.
Below shows the running figure:



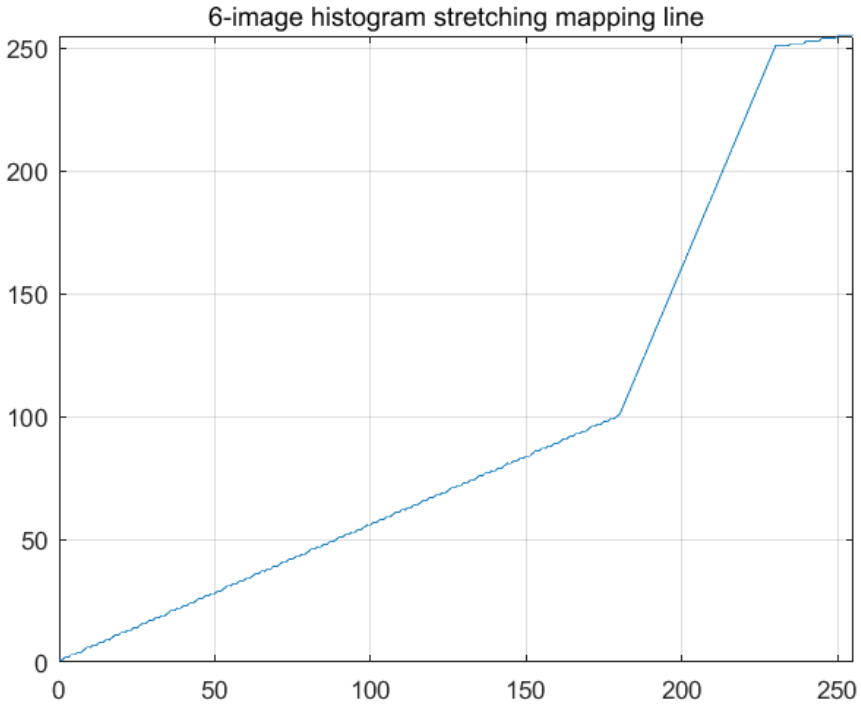
4) See code file Solution_1.m.
Below shows the running figure:



5) See code file Solution_1.m.
Below shows the running figure:



6) See code file imhiststretch.m.
Below shows the running figure:



7) See code file Solution_1.m.
By inspecting the input image and its histogram, I observed that a majority of pixels fall into the range [180, 230], and in order to achieve the best quality which means stretch the image contrast, I decided to expand this range to a much wider range, so I choose [100, 250] range. I map input range [180, 230] to output range [100, 250].

8) See code file Solution_1.m.
Below shows the running figure: