نام و شماره دانشجویی: فاطمه توکلی ۴۰۰۱۳۱۰۱۶

نام درس : پردازش رقمی تصویر

تمرین شماره ۲

.1

a. داریم:

L-1 = $2^3 - 1 = 7$

2	1	2	1	2	5	6	6
3	3	2	3	2	4	5	5
2	2	4	2	1	4	5	5
3	2	4	3	4	2	4	4
3	3	5	4	1	1	2	4
2	3	5	1	1	1	1	4
2	2	3	3	2	3	3	4
5	5	4	4	5	6	5	6

.b

101	110	101	110	101	010	001	001
100	100	101	100	101	011	010	010
101	101	011	101	110	011	010	010
100	101	011	100	011	101	011	011
100	100	010	010	110	110	101	011
101	100	010	110	110	110	110	011
101	101	100	100	101	100	100	011
010	010	011	011	010	001	010	001

K=0(MSB)

1	0	1	0	1	0	1	1
0	0	1	0	1	1	0	0
1	1	1	1	0	1	0	0
0	1	1	0	1	1	1	1
0	0	0	0	0	0	1	1
1	0	0	0	0	0	0	1
1	1	0	0	1	0	0	1
0	0	1	1	0	1	0	1

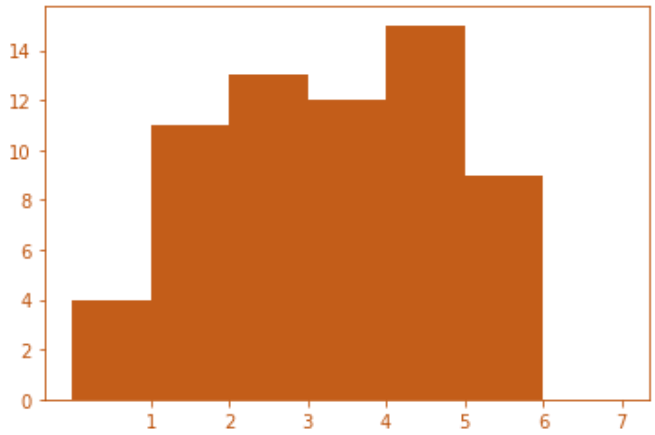
K=1

0	1	0	1	0	1	0	0
0	0	0	0	0	1	1	1
0	0	1	0	1	1	1	1
0	0	1	0	1	0	1	1
0	0	1	1	1	1	0	1
0	0	1	1	1	1	1	1
0	0	0	0	0	0	0	1
1	1	1	1	1	0	1	0

K=2

1	1	1	1	1	0	0	0
1	1	1	1	1	0	0	0
1	1	0	1	1	0	0	0
1	1	0	1	0	1	0	0
1	1	0	0	1	1	1	0
1	1	0	1	1	1	1	0
1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0

.c



.d

با در نظر گرفتن حد آستانه ۵ داریم:

7	7	7	7	7	0	0	0
0	0	7	0	7	0	0	0
7	7	0	7	7	0	0	0
0	7	0	0	0	7	0	0
0	0	0	0	7	7	7	0
7	0	0	7	7	7	7	0
7	7	0	0	7	0	0	0
0	0	0	0	0	0	0	0

.e

تک تک اعداد را در $\frac{255}{7}$ ضرب می کنیم:

182	219	182	219	182	73	36	36
146	146	182	146	182	109	73	73
182	182	109	182	219	109	73	73
146	182	109	146	109	182	109	109
146	146	73	109	219	219	182	109
182	146	73	219	219	219	218	109
182	182	146	146	182	182	146	109
73	73	109	109	73	36	73	36

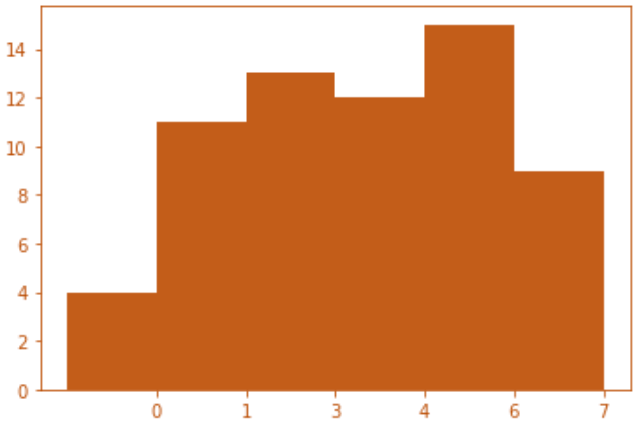
f. ابتدا اعداد را به فضای جدید نگاشت میکنیم:

value	frequency	CDF	S	[S]
1	4	1/16	0.4375	0
2	11	15/64	1.640625	1
3	13	14/32	3.0625	3
4	12	5/8	4.375	4
5	15	55/64	6.015625	6
6	9	1	7	7
7	0	1	7	7

حال ماتریس را به فضای جدید انتقال می دهیم:

6	7	6	7	6	1	0	0
4	4	6	3	6	3	1	1
6	6	3	6	7	3	1	1
4	6	3	4	3	6	3	3
4	4	1	3	7	7	6	3
6	4	1	7	7	7	7	3
6	6	4	4	6	4	4	3
2	2	3	3	1	0	1	0

حال هیستوگرام جدید به صورت زیر است:

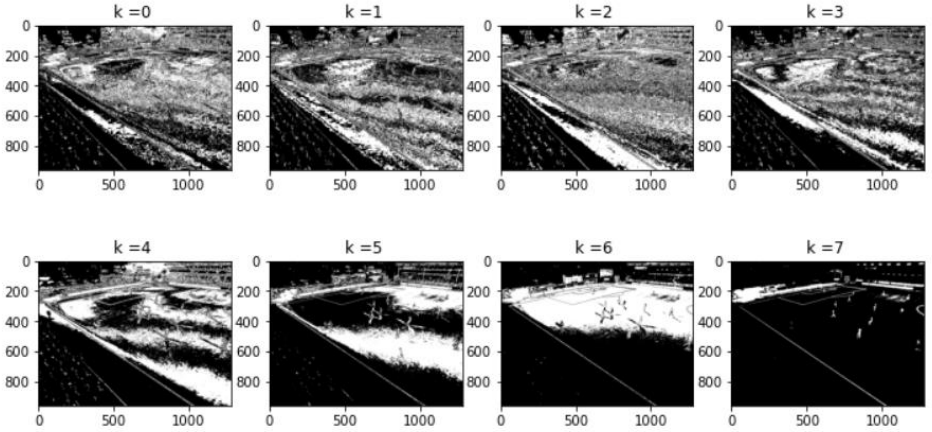


g. ابتدا اعداد را به فضای جدید نگاشت میکنیم:

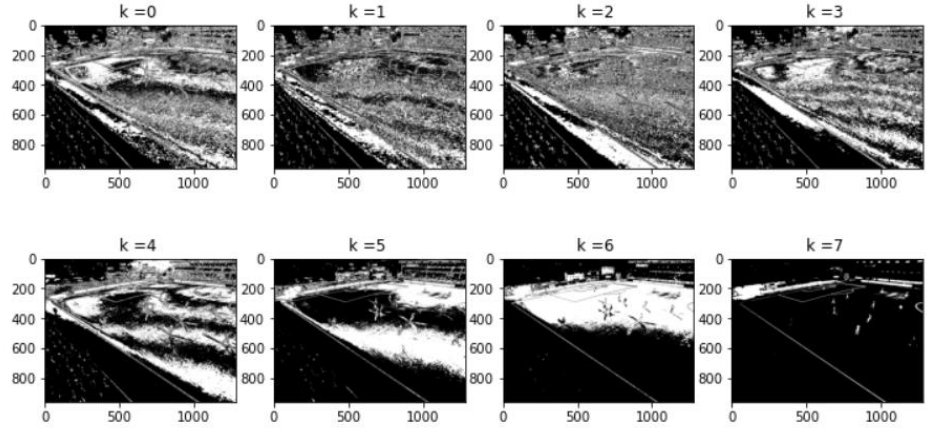
value	frequency	CDF	S	[S]
0	0	0	0	0
1	0	0	0	0
2	512	1/2	3.5	3
3	0	1/2	3.5	3
4	0	1/2	3.5	3
5	0	1/2	3.5	3
6	512	1	7	7
7	0	1	7	7

2. تمامی کدها در پیوست آورده شده‌است.

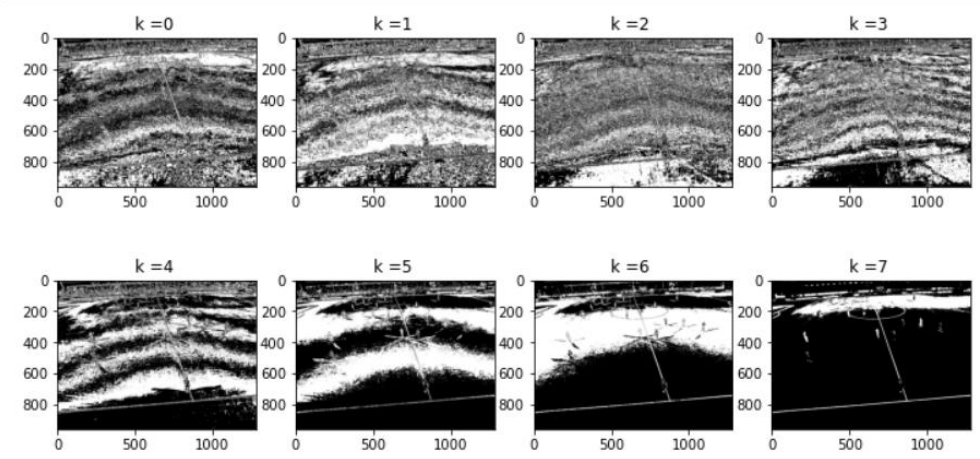
a. تابع خواسته شده پیاده‌سازی شده و نتایج بر روی عکس های داده شده به صورت زیر می‌باشد:



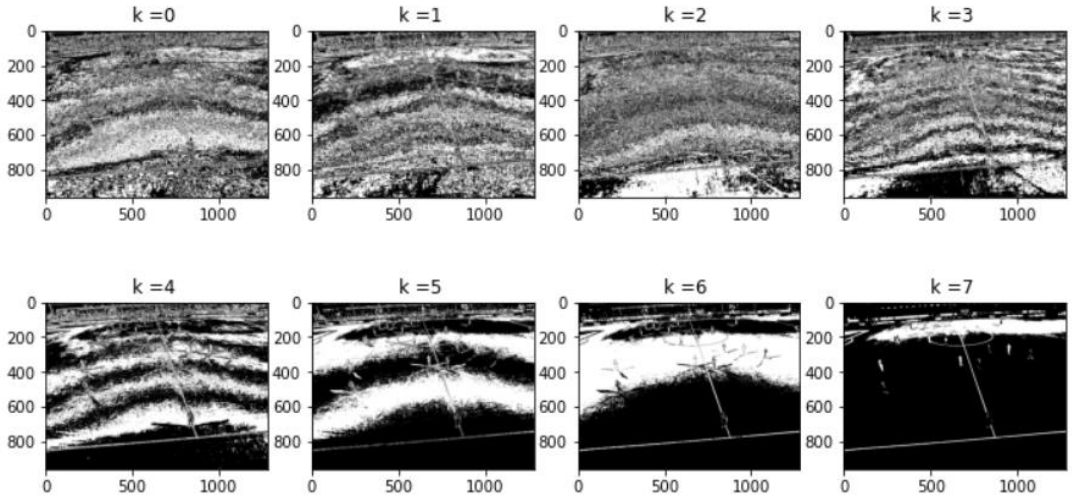
(1)



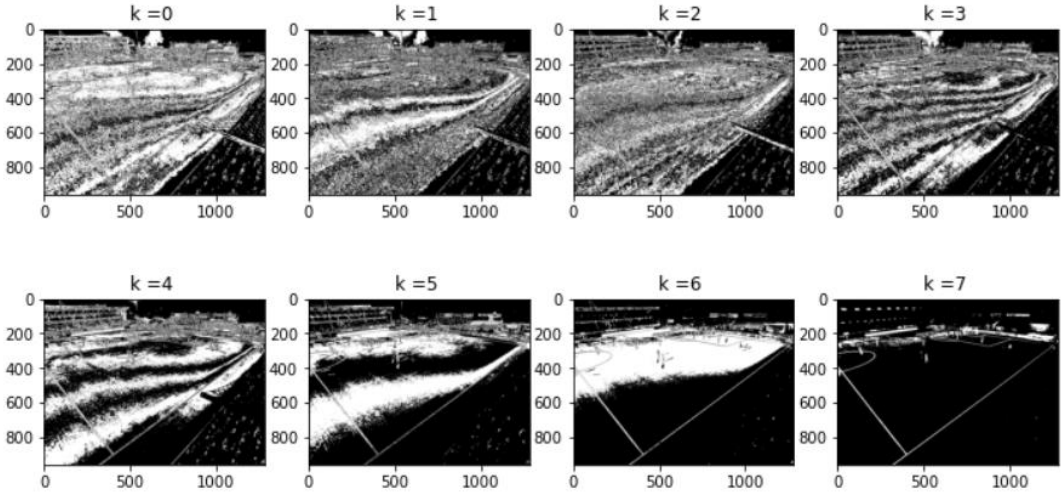
(2)



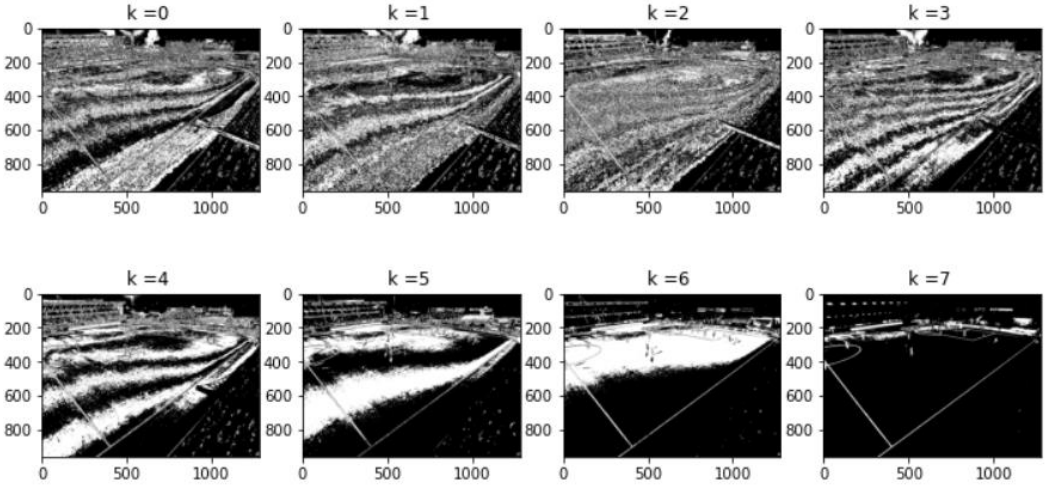
(3)



(4)

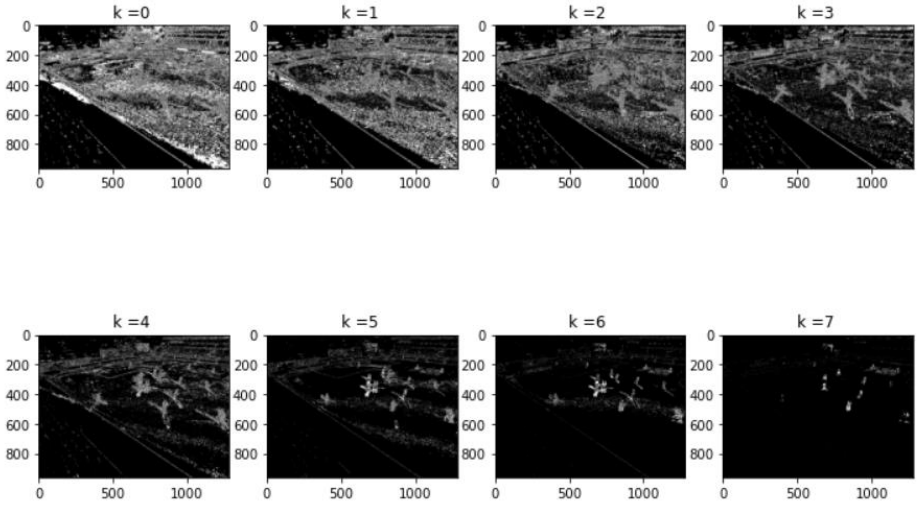


(5)

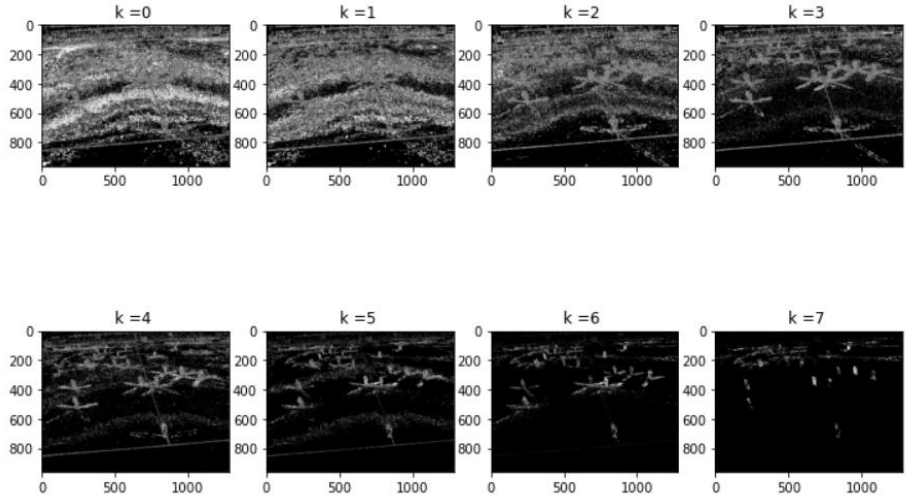


(6)

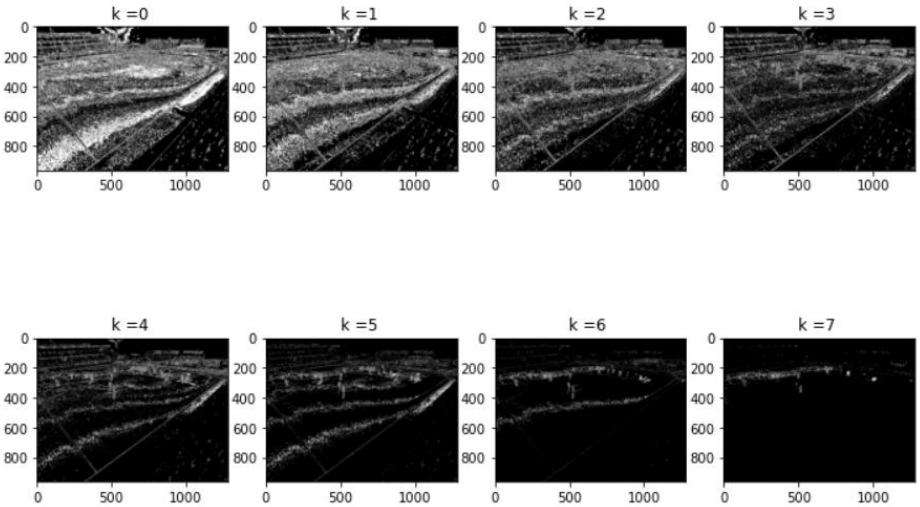
b. نتیجه اعمال xor پیاده شده برروی عکس ها به صورت دو به دو :



(1)

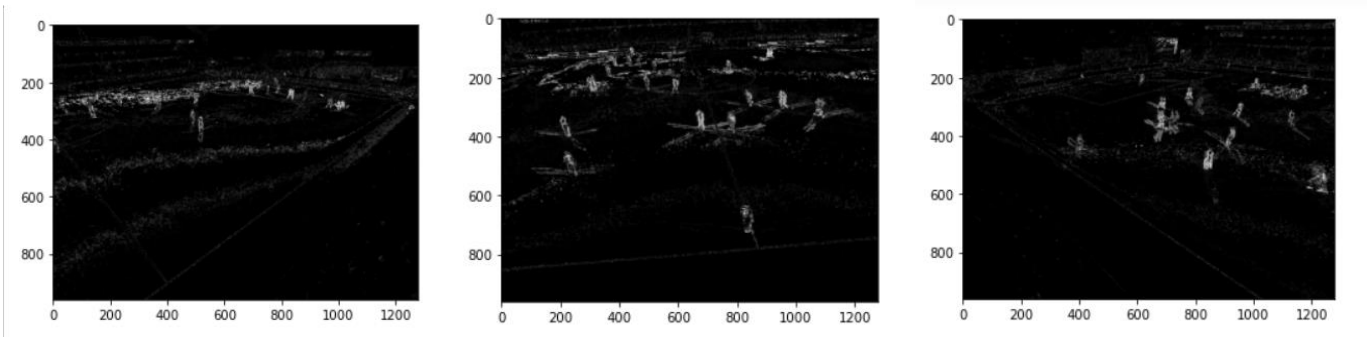


(2)

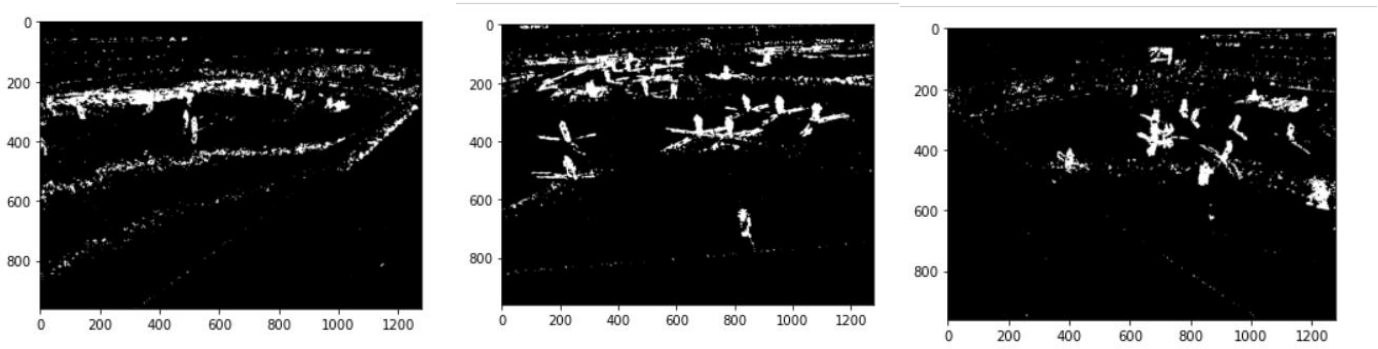


(3)

c. نتایج تغییر قاب:

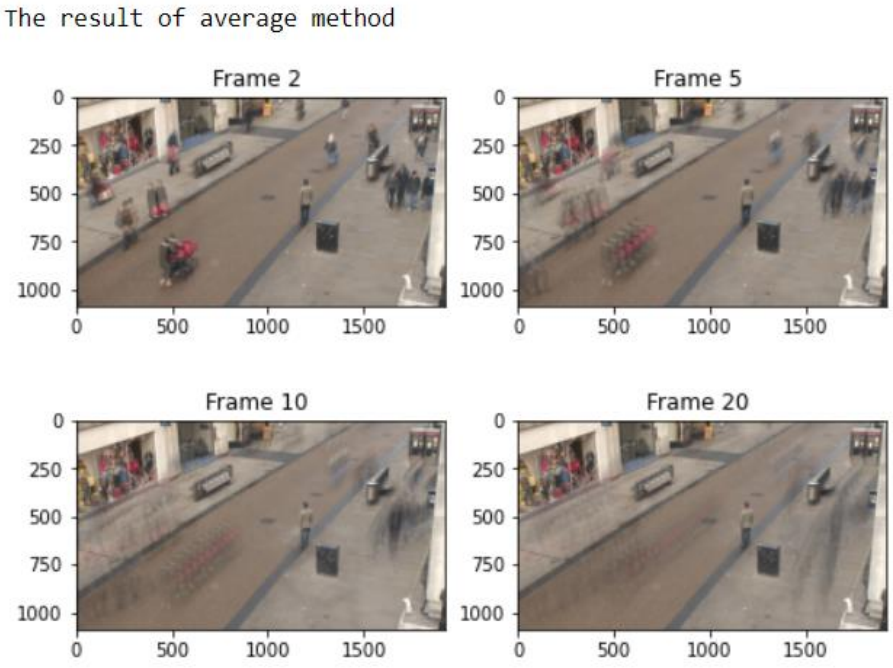


d. برای بهبود خروجیهای قسمتc ، از یک فیلتر میانه 9*9 و threshold با مقدار ۳۰ انجام شده است. اجسام دورتر نسبت به زاویه دوربین هنگام اعمال فیلتر همراه نویز ها حذف شده‌اند و نتایج به صورت زیر می‌باشد:

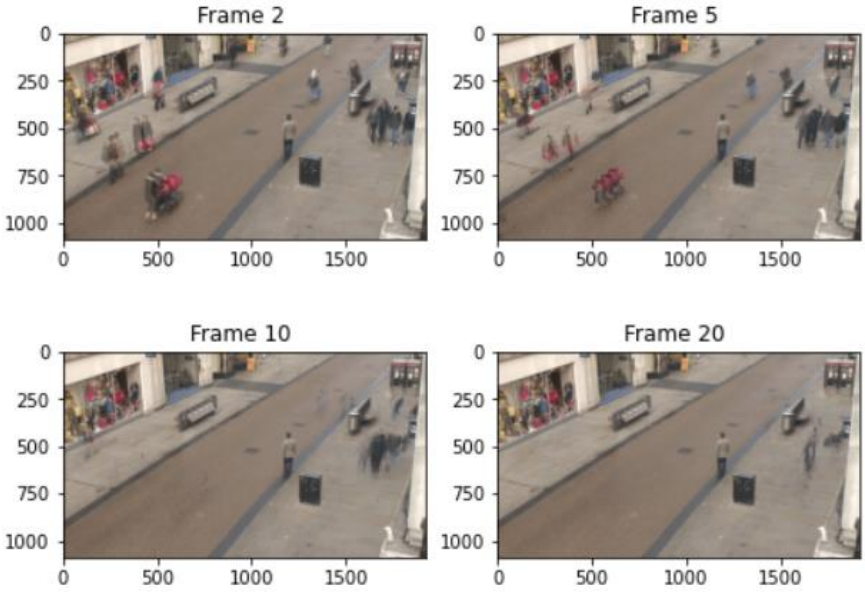


.3

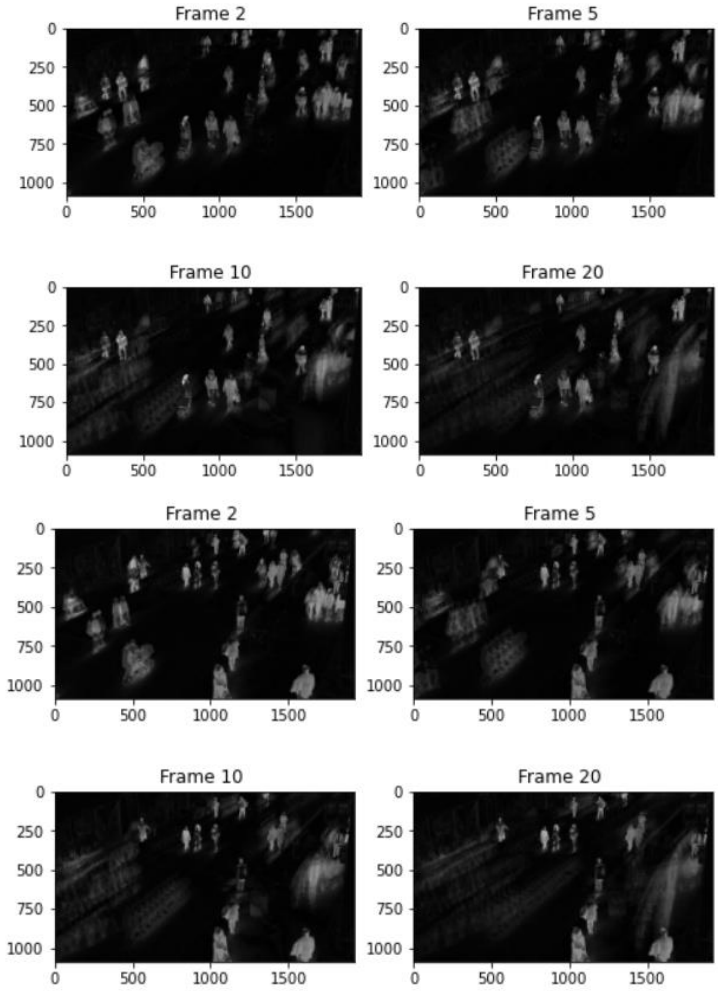
a. نتیجه اعمال دو متد خواسته شده:



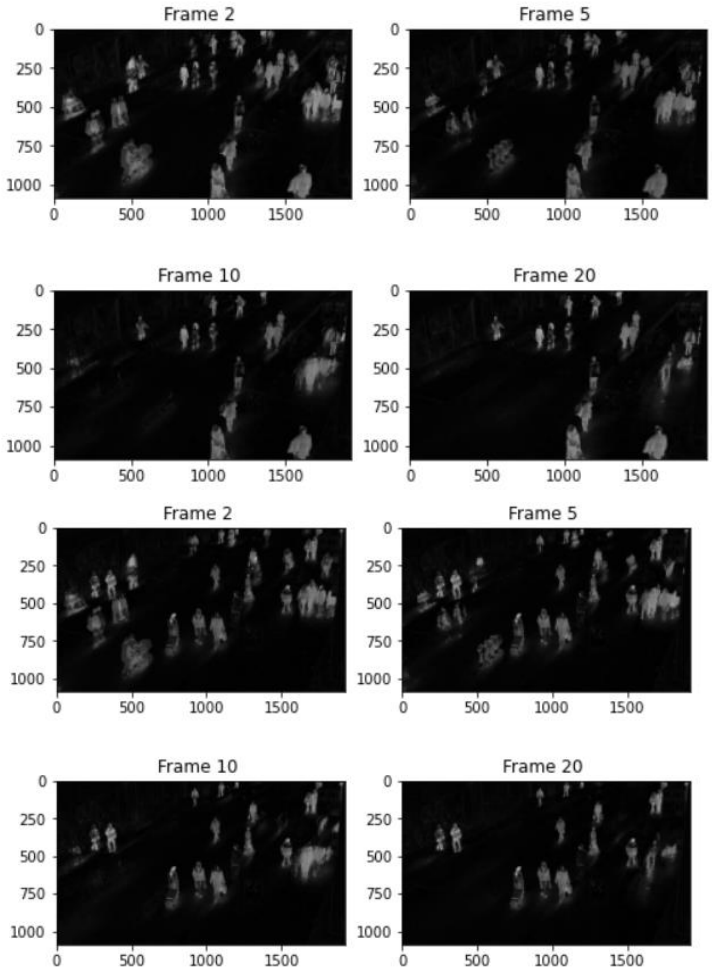
The result of median method



b. نتیجه تفاضل خواسته شده و تکرار نتایج مشابه قسمت قبل:

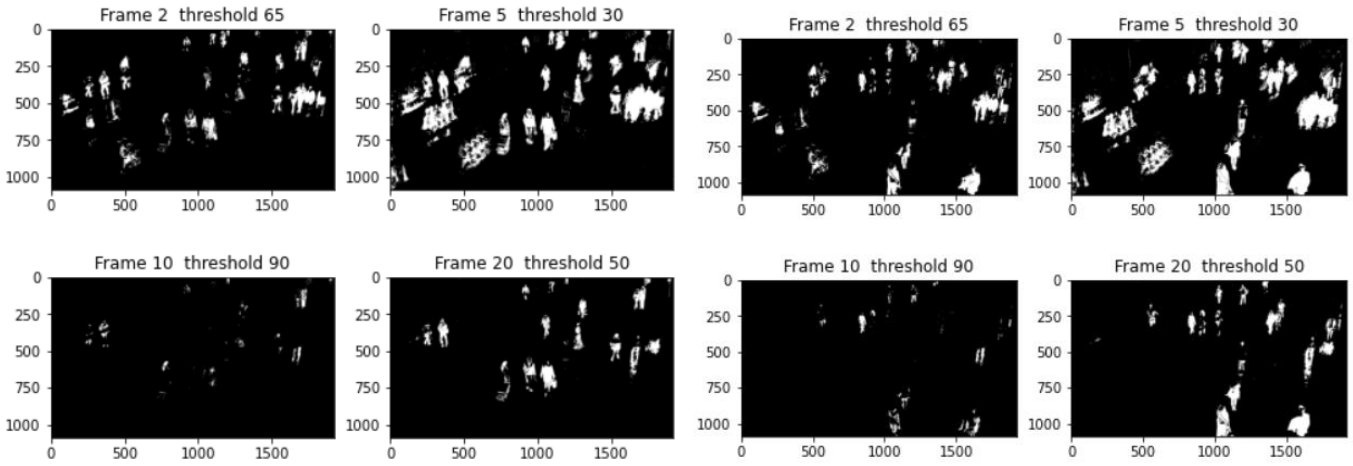


Result of average method

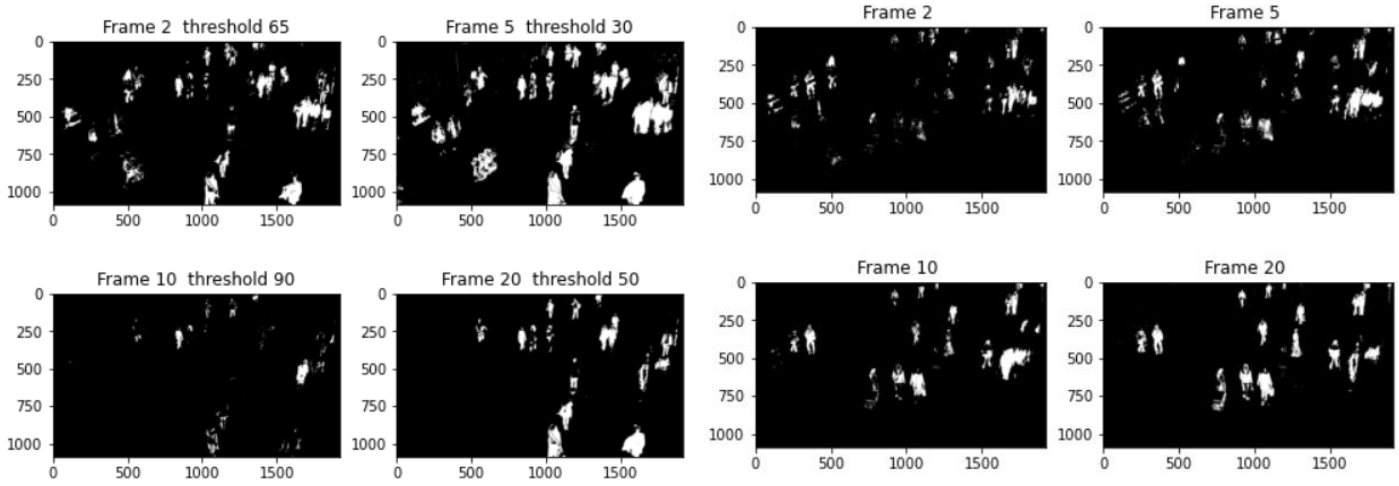


Result of median average

.c

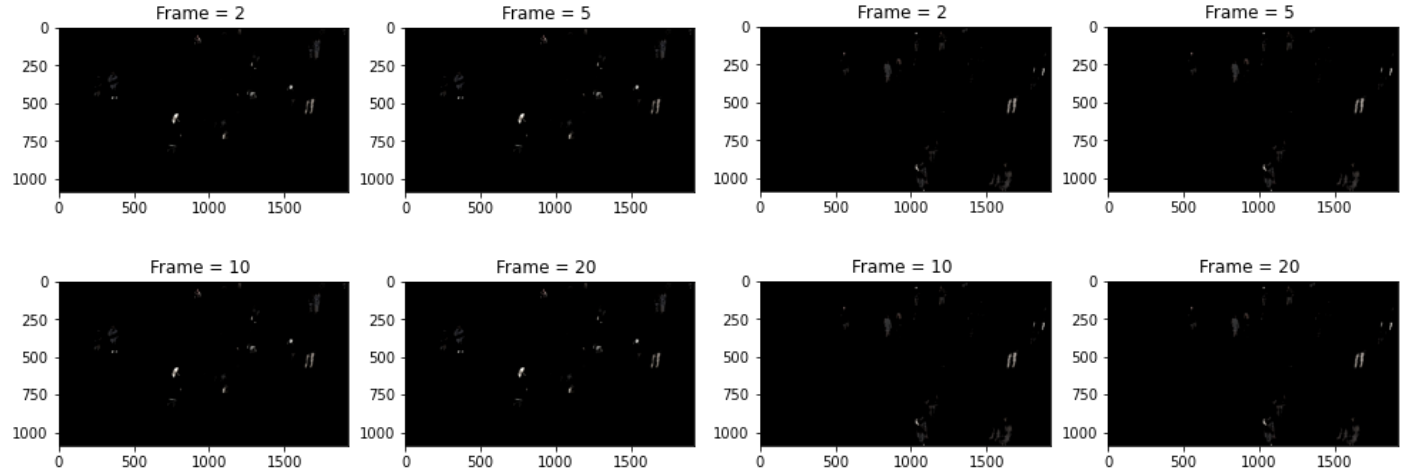


Result of average method with different threshold

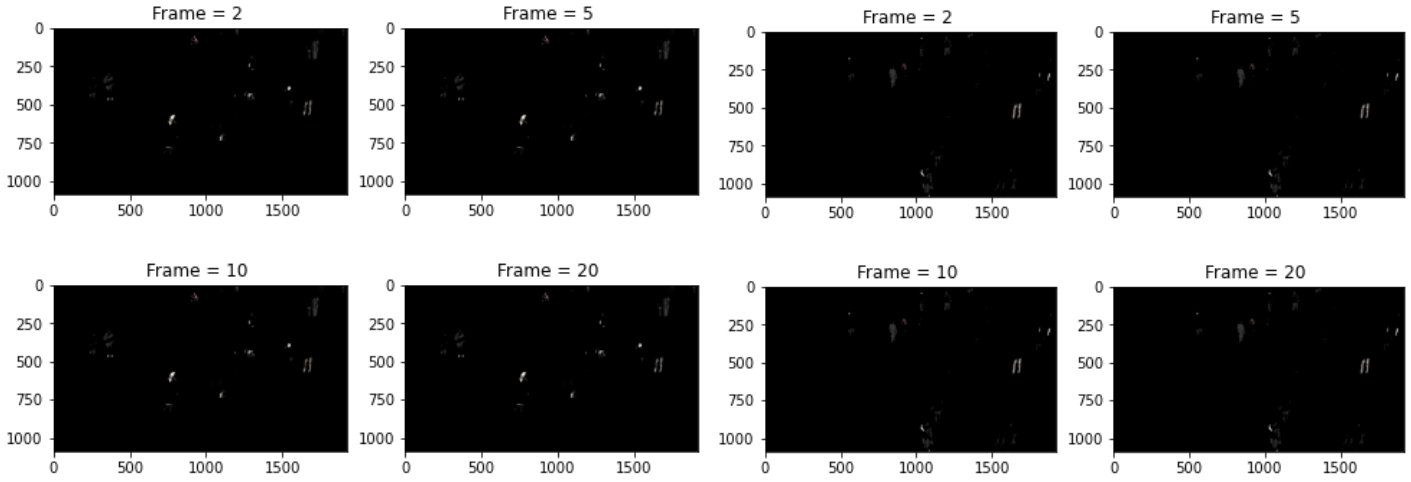


Result of median method with different threshold

.d



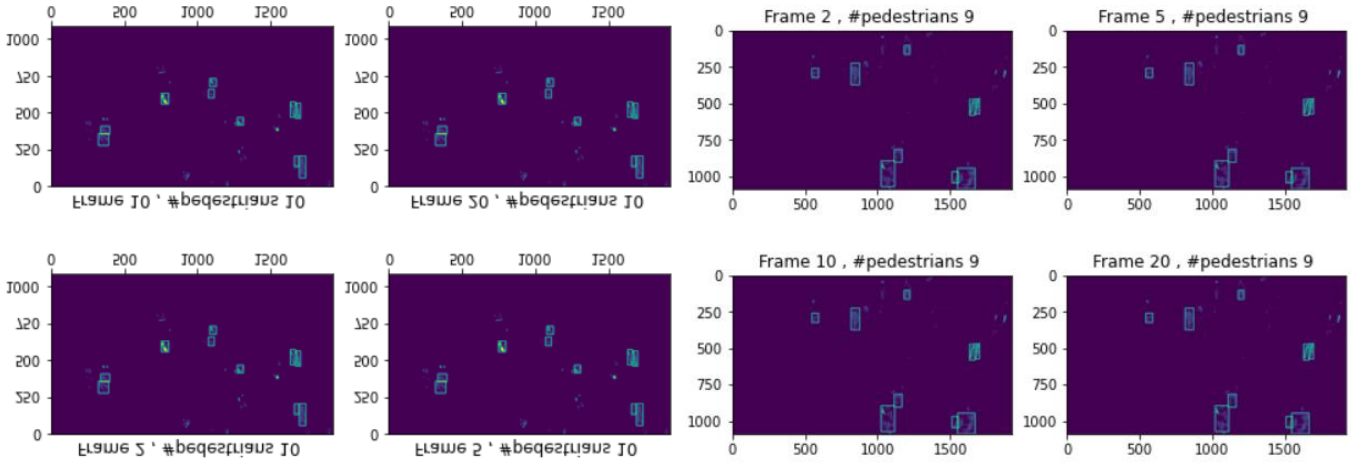
Result of the average method



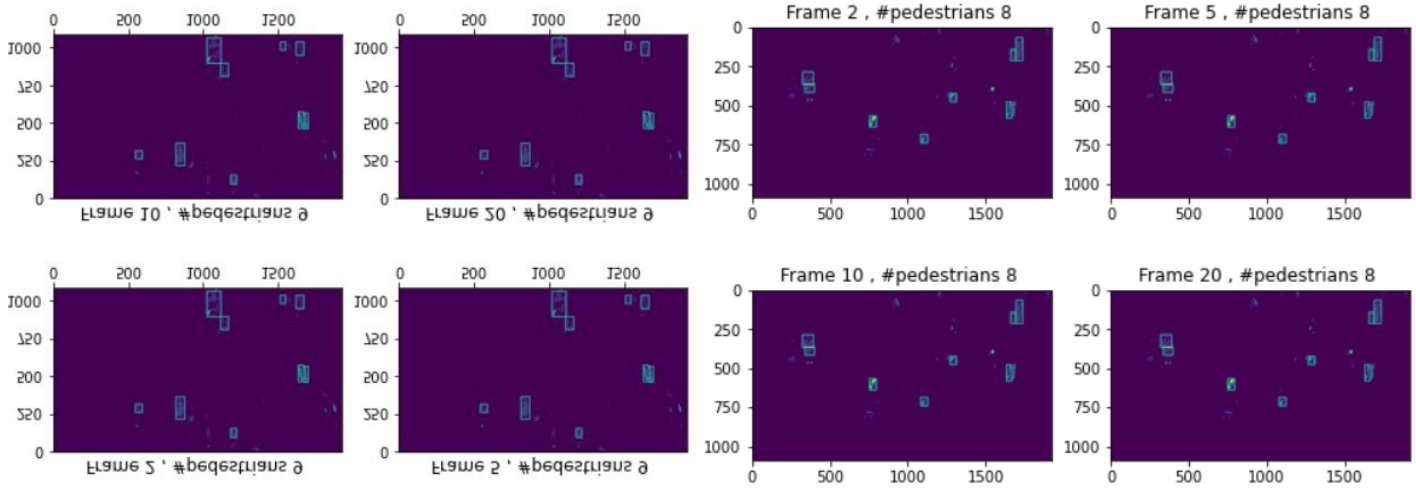
Result of the median method

e. در هر سئوال پاسخ داده شده است

.f



Result of the average method

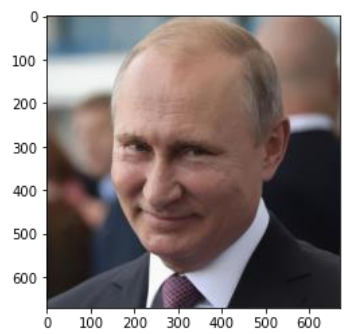


Result of the median method

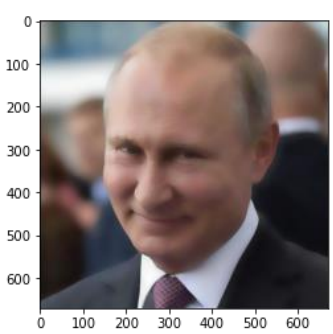
.a .4

```
def designed_bilateral_filter(image,d,sigma_c,sigma_d):
    b,g,r = cv2.split(image)
    bilatb=np.zeros(b.shape)
    bilatg=np.zeros(g.shape)
    bilatr=np.zeros(r.shape)

    m= int((d-1)/2)
    for i in range(m,b.shape[0]-m):
        for j in range(m,b.shape[1]-m):
            Wp1=0
            Wp2=0
            Wp3=0
            for k in range(i-m,m+i+1):
                for l in range(j-m,m+j+1):
                    Wp1+=weight(b,i,j,k,l,sigma_c,sigma_d)
                    Wp2+=weight(g,i,j,k,l,sigma_c,sigma_d)
                    Wp3+=weight(r,i,j,k,l,sigma_c,sigma_d)
                    bilatb[i,j]+=b[k,l]*weight(b,i,j,k,l,sigma_c,sigma_d)
                    bilatg[i,j]+=g[k,l]*weight(g,i,j,k,l,sigma_c,sigma_d)
                    bilatr[i,j]+=r[k,l]*weight(r,i,j,k,l,sigma_c,sigma_d)
                    if (k==m+i) and (l==m+j):
                        bilatb[i,j]=int(round(bilatb[i,j]/Wp1))
                        bilatg[i,j]=int(round(bilatg[i,j]/Wp2))
                        bilatr[i,j]=int(round(bilatr[i,j]/Wp3))
    for i in range(0,m):
        for j in range(0,b.shape[1]):
            bilatb[i,j]=b[i,j]
            bilatg[i,j]=g[i,j]
            bilatr[i,j]=r[i,j]
    for i in range(b.shape[0]-m-1,b.shape[0]):
        for j in range(0,b.shape[1]):
            bilatb[i,j]=b[i,j]
            bilatg[i,j]=g[i,j]
            bilatr[i,j]=r[i,j]
    for i in range(0,b.shape[0]):
        for j in range(0,m):
            bilatb[i,j]=b[i,j]
            bilatg[i,j]=g[i,j]
            bilatr[i,j]=r[i,j]
    for i in range(0,b.shape[0]):
        for j in range(b.shape[1]-m-1,b.shape[1]):
            bilatb[i,j]=b[i,j]
            bilatg[i,j]=g[i,j]
            bilatr[i,j]=r[i,j]

    bilat1 = np.zeros(image.shape)
    bilat1[..., 0] = bilatb # Red Matrix
    bilat1[..., 1] = bilatg # Green Matrix
    bilat1[..., 2] = bilatr # Red Matrix
    return bilat1

def distance(x, y, p, q):
    return np.sqrt((x-p)**2 + (y-q)**2)

def gaussian(x, sigma):
    return (1.0 /np.sqrt(2 * math.pi * (sigma ** 2)))* math.exp(- (x ** 2) / (2 * sigma ** 2))

def weight(image,i,j,k,l,sigma_c,sigma_d):
    fg = gaussian(image[k][l] - image[i][j], sigma_c)
    gs = gaussian(distance(i, j, k, l), sigma_d)
    w = fg * gs
    return w
```

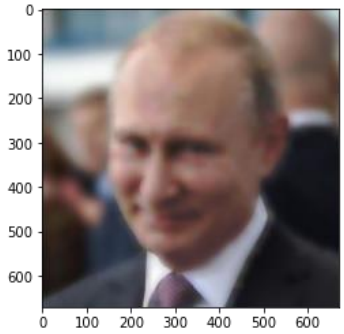
.b



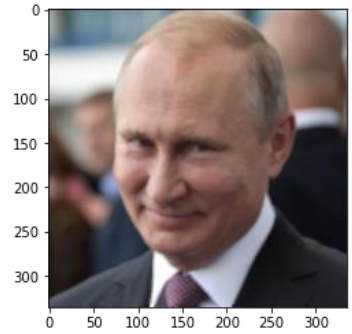
Spatial =30,range = 10



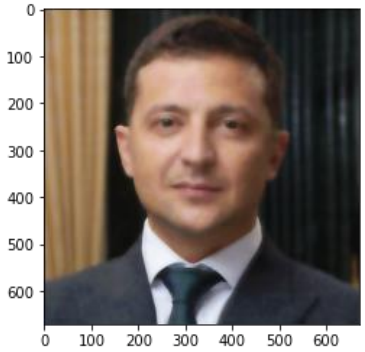
Spatial = 10 , range= 30



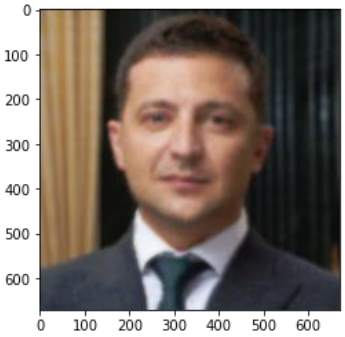
spatial = 10, range =100



spatial = 100, range =30

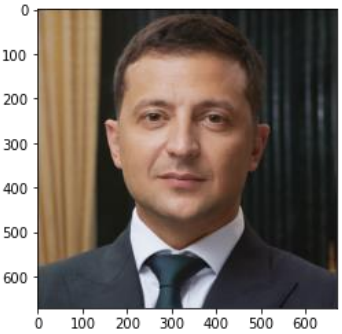


Spatial = 5 , range = 50



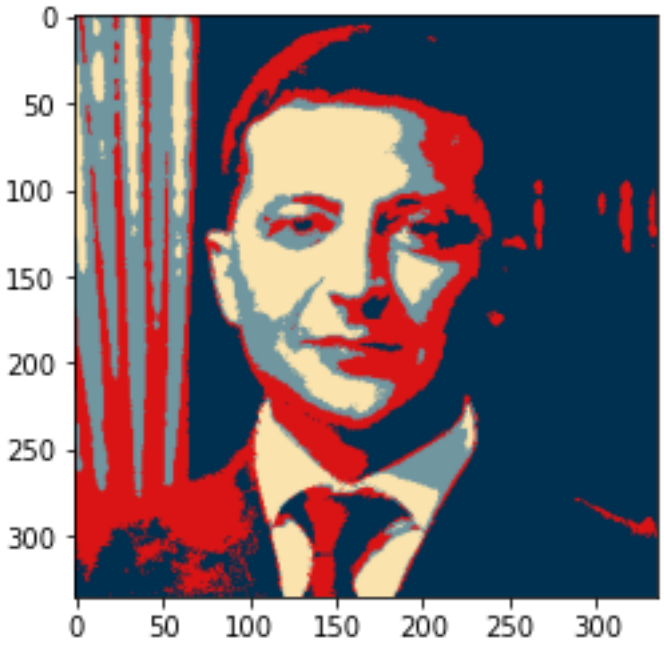
spatial =5, range = 150

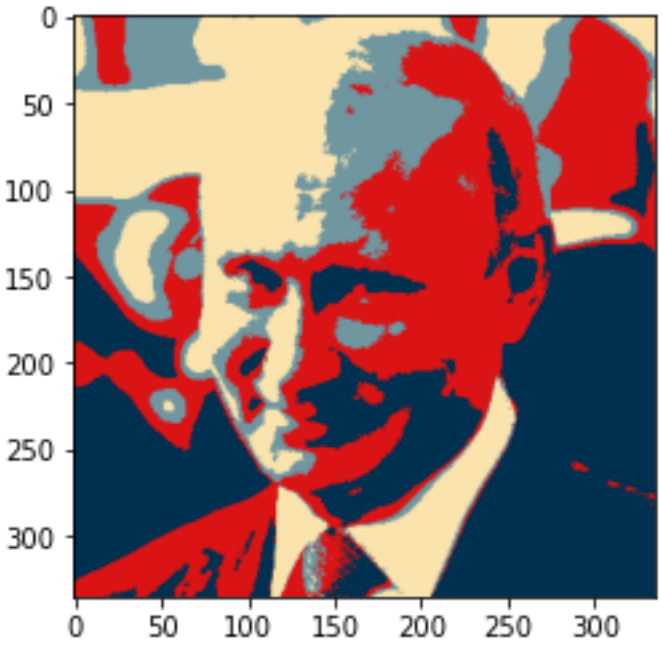
spatial = 50,range = 15



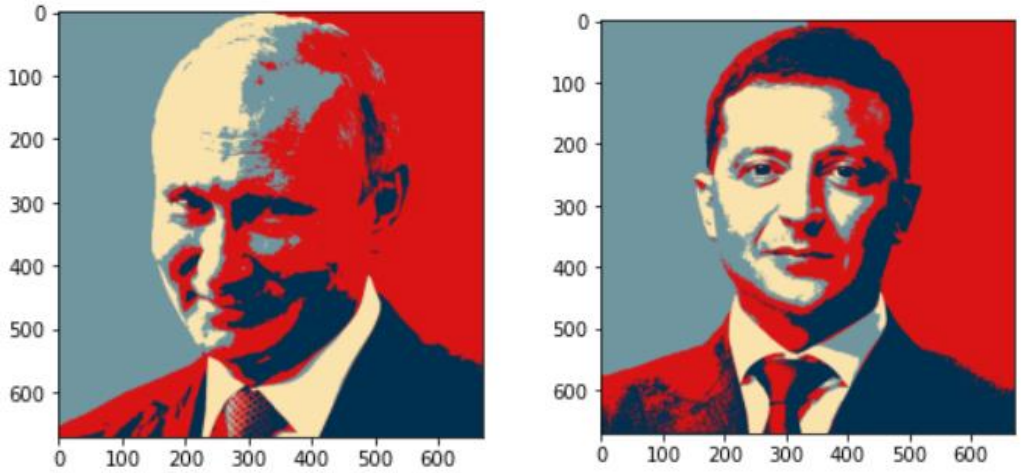
spatial =150,range =15

.c





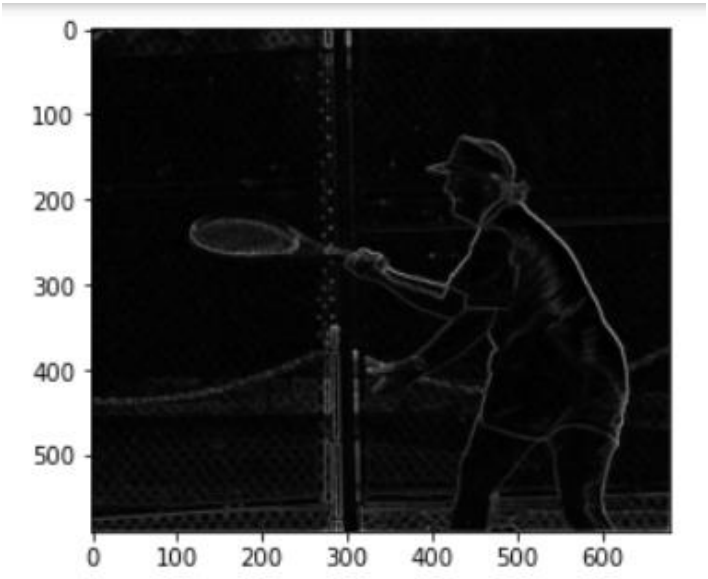
d. نتیجه اعمال ماسک داده شده و تقسیم پس‌زمینه:

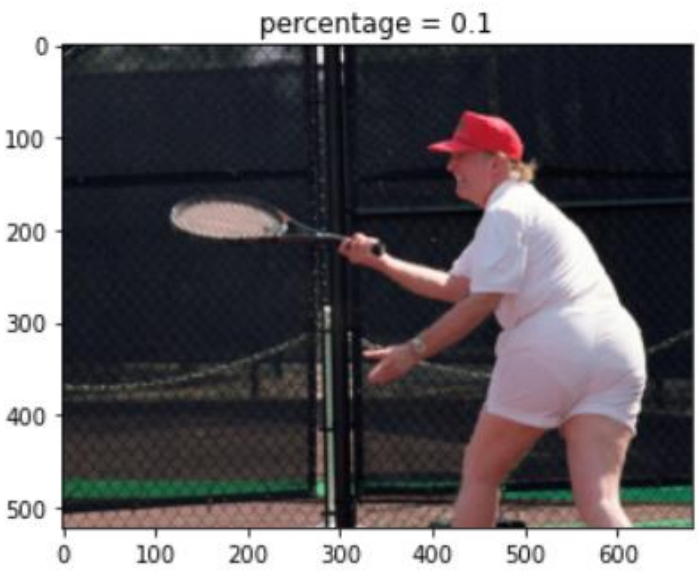


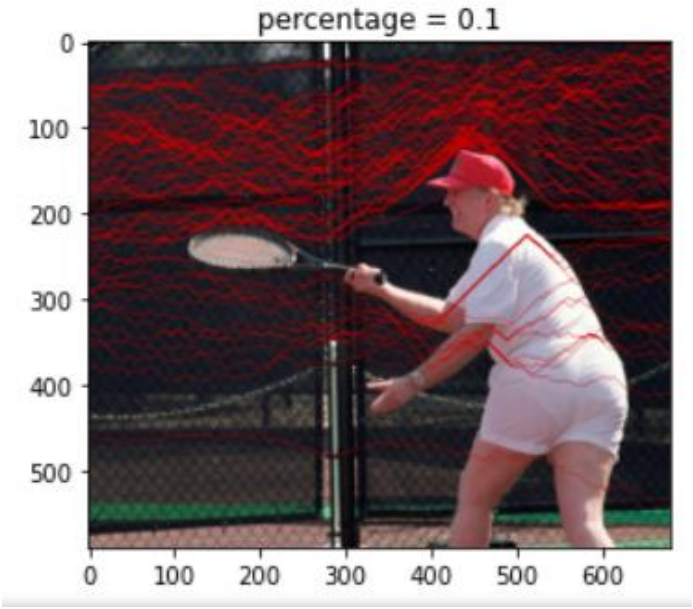
e. آستانه شدت تصویر به صورت دستی در یک مقدار مشخص یا به طور خودکار توسط یک برنامه تنظیم می شود. پیکسل های زیر آن مقدار آستانه تعیین شده به یک رنگ تعریف شده مثلا سیاه تبدیل می شوند (مقدار ب صفر) و پیکسل های بالاتر از مقدار آستانه به رنگ دیگر تعریف شده مثلا سفید (مقدار بیت یک) تبدیل می شوند. و مقدار استانه در تقسیم‌بندی عکس موثر می‌باشد

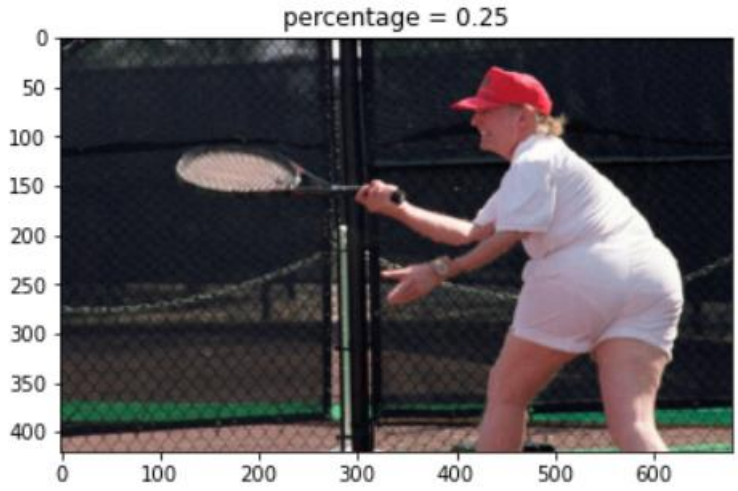
.5

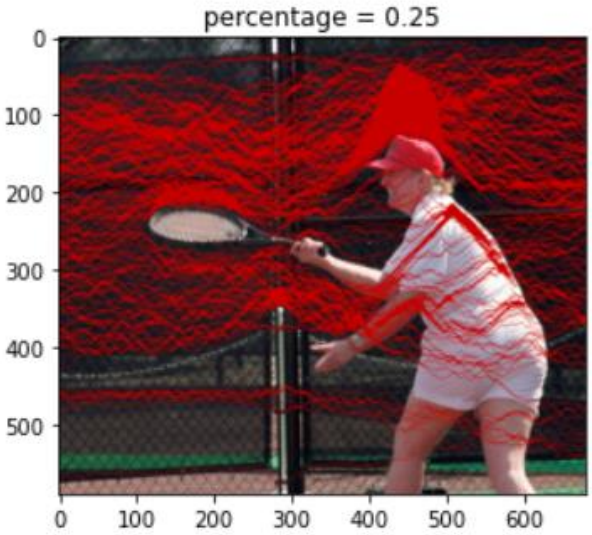
a. کاهش عرض عکس a :

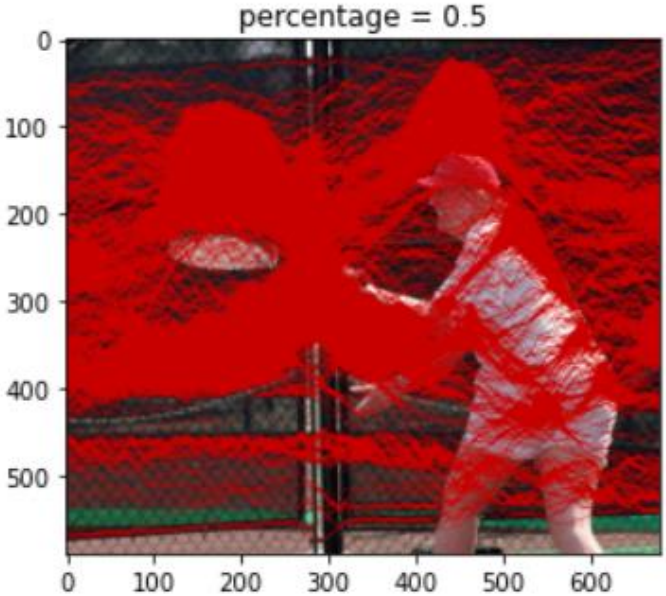


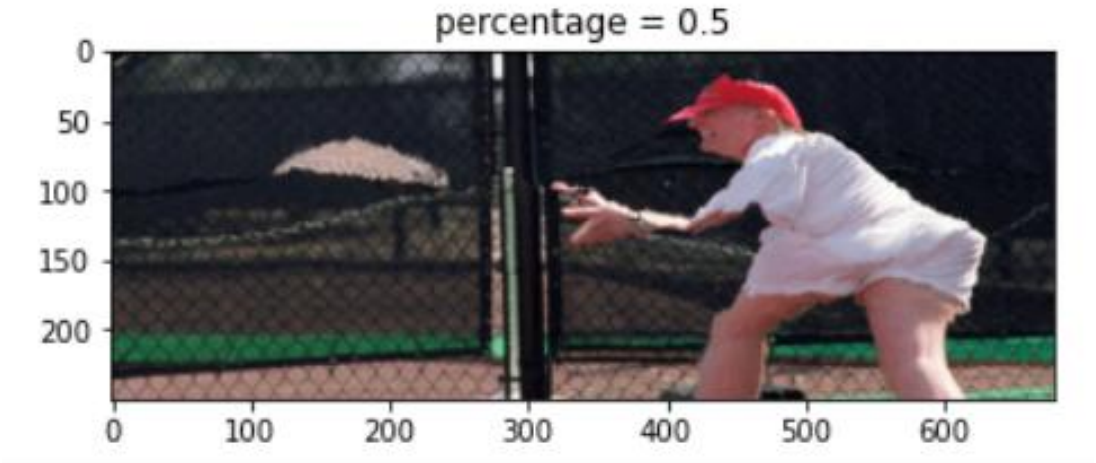

percentage = 0.1


percentage = 0.1

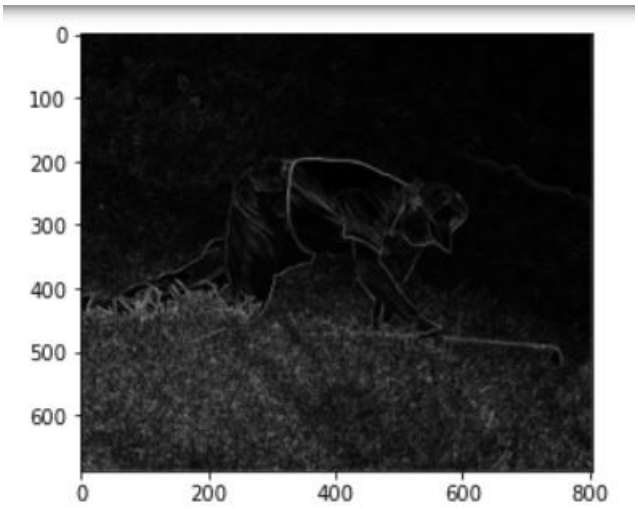

percentage = 0.25

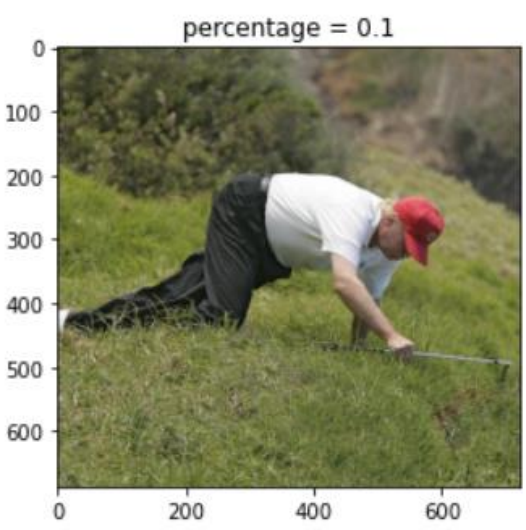

percentage = 0.25

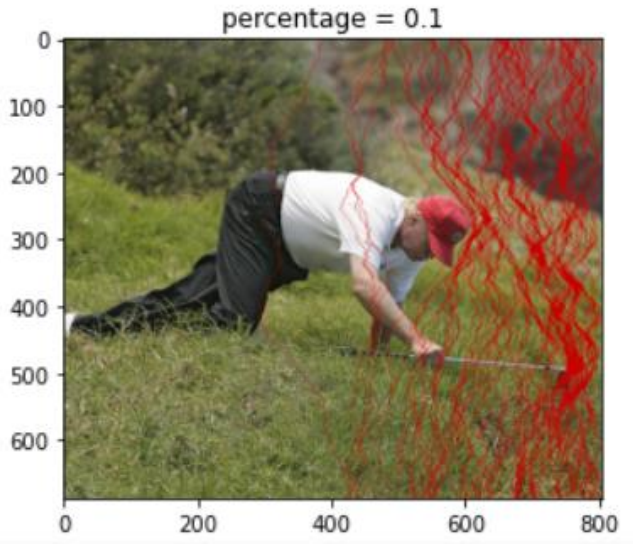

percentage = 0.5



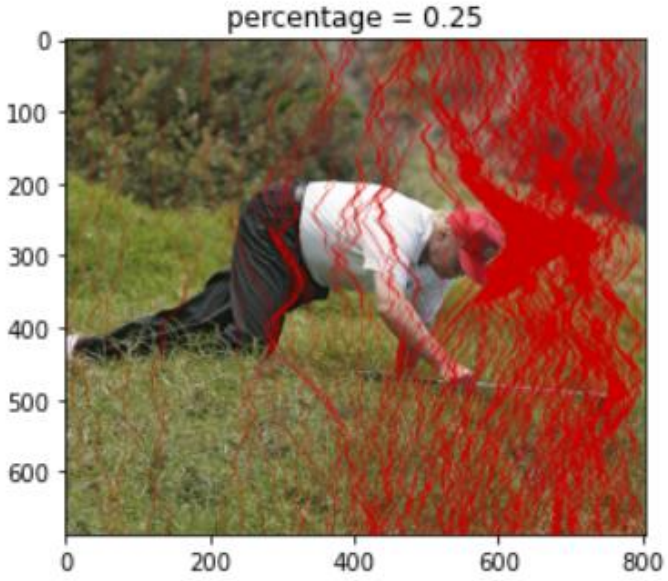
b. حاصل کاهش طول :



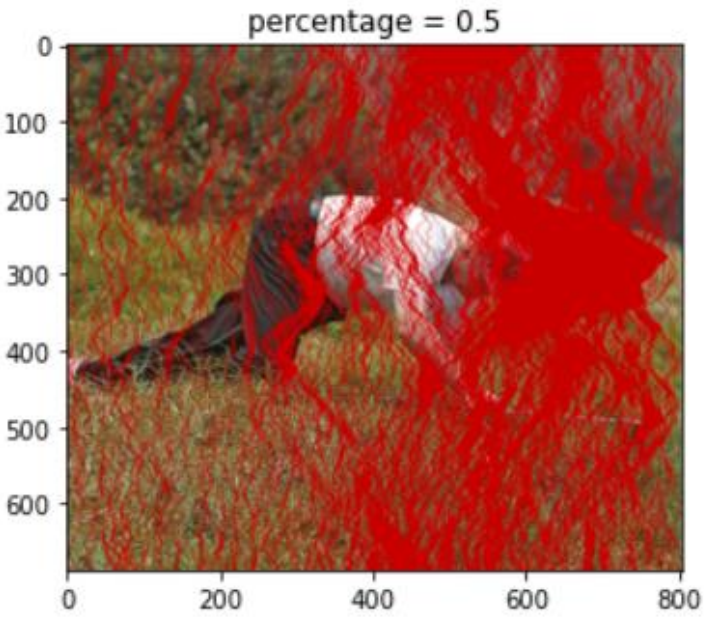




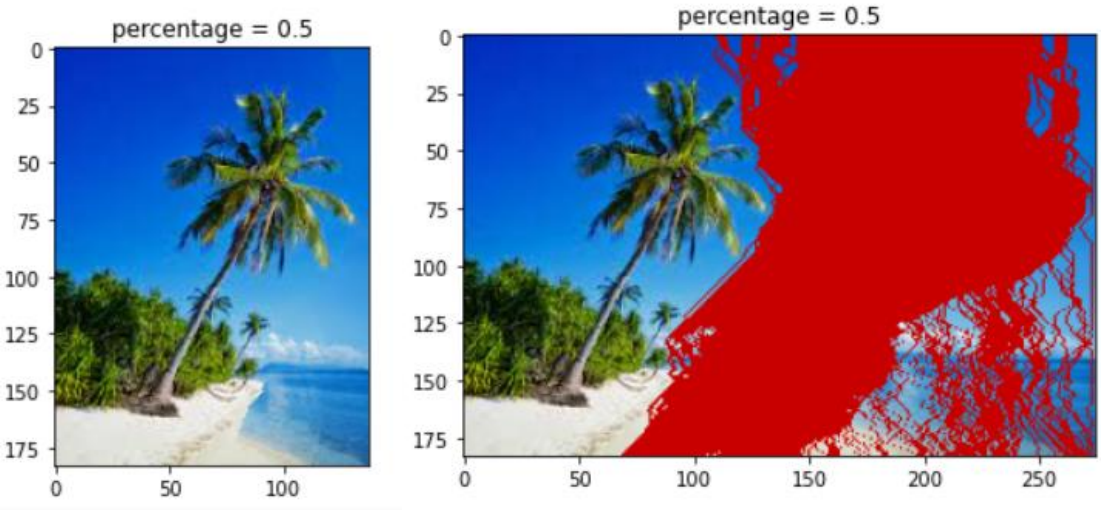

percentage = 0.25
0
100
200
300
400
500
600
0
200
400
600


percentage = 0.25
0
100
200
300
400
500
600
0
200
400
600
800

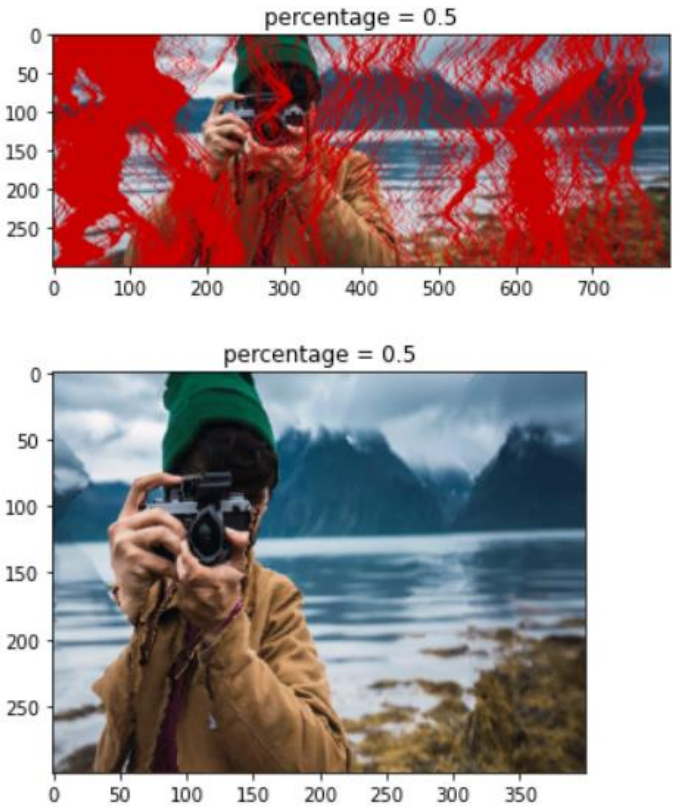

percentage = 0.5
0
100
200
300
400
500
600
0
200
400


percentage = 0.5
0
100
200
300
400
500
600
0
200
400
600
800

c. نتیجه با کاهش عمودی :



d. با کاهش عمودی:



6.

a. خیر نمی‌توان فیلتر بزرگتر از تصویر زیرا جواب کانولوشن جواب نمیدهد. اما ممکن است بتوان فیلتر را کوچک‌تر یا هم‌اندازه با تصویر کرد و جواب نهایی را به دست آورد.

b. بله، پس از اعمال متوالی این فیلتر، لبه ها smooth تر و نویز کمتر می‌شود. در نهایت یک تصویر تار خواهیم داشت که اعمال بیشتر این فیلتر تغییری در آن ایجاد نخواهد کرد.

d. برای اعمال دو فیلتر عمودی و افقی با طول n ، در هر نقطه به $2n$ عمل ضرب نیاز داریم. در طرف مقابل برای اعمال فیلتر مربعی به ضلع n ، به n^2 عمل ضرب نیاز داریم؛ بنابراین حالت اول بهینه‌تر است.

e.میانگین‌گیری تصویر یک تکنیک پردازش تصویر دیجیتال است که اغلب برای بهبود تصاویر ویدیویی که توسط نویز تصادفی خراب شده‌اند، استفاده می‌شود. این الگوریتم با محاسبه میانگین یا میانگین حسابی مقادیر intensity برای هر موقعیت پیکسل در مجموعه ای از تصاویر گرفته شده از همان صحنه عمل می کند.