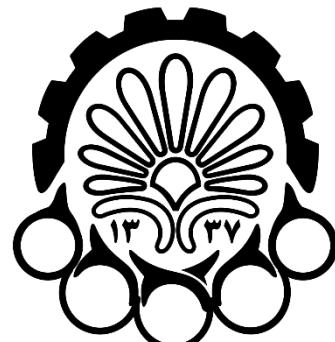


به نام خدا



دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

تمرین درس پردازش تصویر-سری دوم

فردين آيار

شماره دانشجویی: ۹۹۱۳۱۰۴۰

استاد: دکتر رحمتی

دانشکده کامپیوتر - زمستان ۹۹

(۱)

(a) با استفاده از فرمول $s = L - 1 - r$ داریم:

6	5	4	4	4	2	1	1
5	5	5	3	1	1	0	0
5	5	4	3	2	0	0	1
5	4	4	4	1	0	1	1
4	3	2	1	2	3	3	2
5	4	2	2	3	3	3	2
5	3	3	3	3	5	4	3
4	3	3	4	4	5	6	3

(b)

k = 2 (MSB)

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

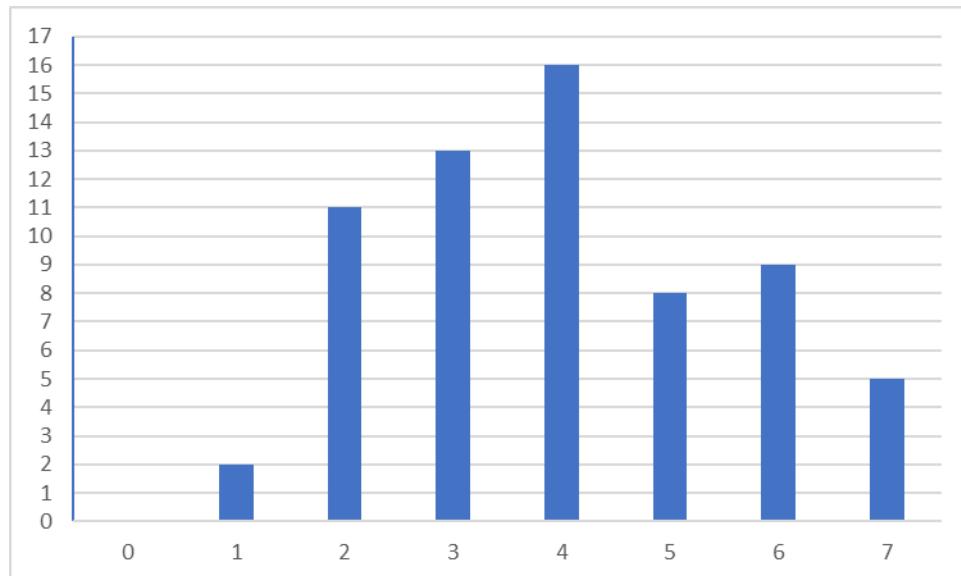
k = 1

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

k = 0

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

(c) هیستوگرام تصویر به شکل زیر است:



(d) با در نظر گرفتن حد آستانه ۵ داریم:

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

(e) اعداد را در $\frac{255}{7}$ ضرب می کنیم.

36	73	109	109	109	182	219	219
----	----	-----	-----	-----	-----	-----	-----

73	73	73	146	219	219	255	255
73	73	109	146	182	255	255	219
73	109	109	109	219	255	219	219
109	146	182	219	182	146	146	182
73	109	182	182	146	146	146	182
73	146	146	146	146	73	109	146
109	146	146	109	109	73	36	146

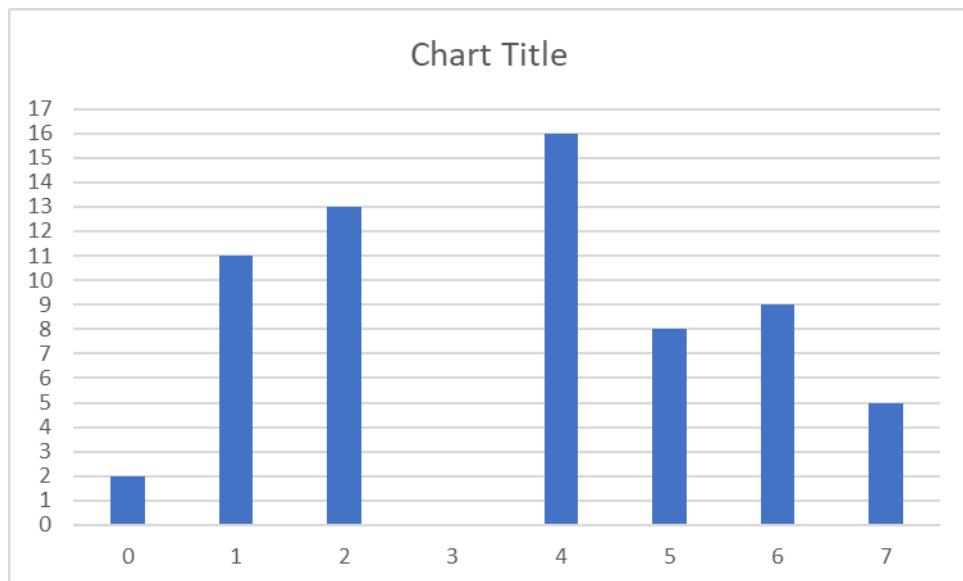
f) ابتدا اعداد را به فضای جدید نگاشت می کنیم.

r	Values	CDF	S	[s]
0	0	0	0	0
1	2	1/32	0.21875	0
2	11	13/64	1.421875	1
3	13	13/32	2.84375	2
4	16	21/32	4.59375	4
5	8	25/32	5.46875	5
6	9	59/64	6.453125	6
7	5	1	7	7

سپس ماتریس را به فضای جدید منتقل می کنیم.

0	1	2	2	2	5	6	6
1	1	1	4	6	6	7	7
1	1	2	4	5	7	7	6
1	2	2	2	6	7	6	6
2	4	5	6	5	4	4	5
1	2	5	5	4	4	4	5
1	4	4	4	4	1	2	4
2	4	4	2	2	1	0	4

در انتهای هیستوگرام تصویر جدید رارسم می کنیم.



(g) ابتدا هیستوگرام داده شده را به فضای جدید منتقل می کنیم.

r	Values	CDF	S	[s]
0	0	0	0	0
1	512	1/2	3.5	3
2	0	1/2	3.5	3
3	0	1/2	3.5	3
4	0	1/2	3.5	3
5	512	1	7	7
6	0	1	7	7
7	0	1	7	7

با توجه به جدول فوق و جدول بخش قبل داریم: (اعداد به کوچکترین عدد بعدی نگاشت می شوند)

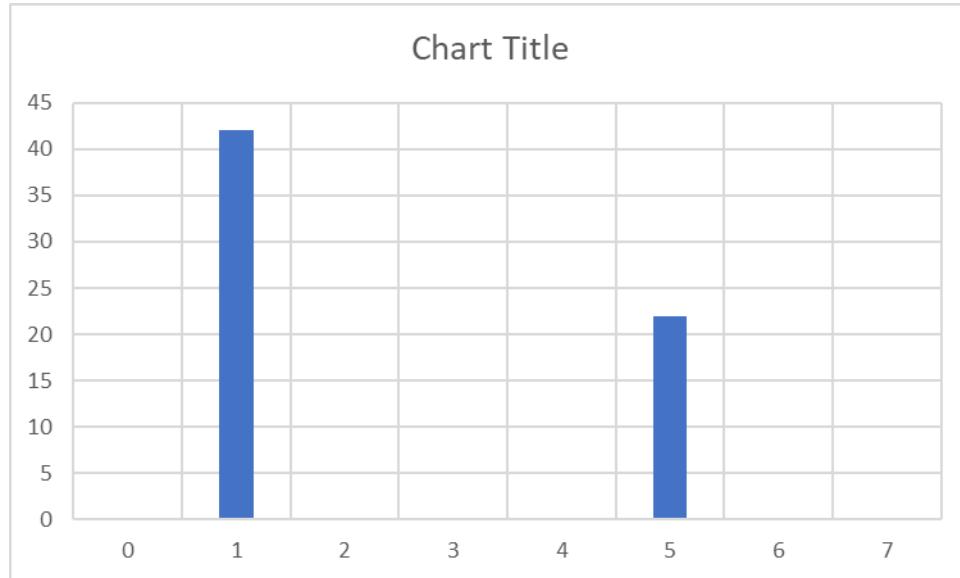
r	S*
0	0
1	1
2	1
3	1
4	1
5	5
6	5
7	5

ماتریس جدید به این صورت است:

1	1	1	1	1	5	5	5
1	1	1	1	5	5	5	5
1	1	1	1	5	5	5	5
1	1	1	1	5	5	5	5

1	1	5	5	5	1	1	5
1	1	5	5	1	1	1	5
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

در نهایت هیستوگرام ماتریس فوق رارسم می کنیم.



(۲)

با توجه به معادله کانولوشن که به صورت زیر تعریف می شود. کافی است هر فیلتر را حول هر دو محور اصلی دوران داده و سپس در تمام نقاط تصویر اعمال کنیم. در ماتریس خروجی اعدادی که با رنگ قرمز مشخص شده اند خارج از محدوده مجاز قرار دارند.

$$f[n_1, n_2] \ast\ast h[n_1, n_2] = \sum_{m_1=-\infty}^{\infty} \sum_{m_2=-\infty}^{\infty} f[m_1, m_2] h[n_1 - m_1, n_2 - m_2]$$

(a) این فیلتر نقاطی که نسبت به همسایگی شان تمایز زیادی دارند مشخص می کند. به بیان دیگر این فیلتر مرزهای موجود در تصویر را مشخص می کند.

-1	-1	-1
-1	8	-1
-1	-1	-1

0	7	7	0	0	0	7	5	7
0	7	0	0	0	0	2	7	5
0	7	0	7	7	1	0	7	0
0	7	0	0	0	0	7	0	0
0	0	7	0	7	2	7	0	0

7	0	7	0	7	0	6	0
7	0	7	0	7	2	0	7
7	7	0	0	0	7	0	7

(b) این فیلتر مرزهای عمودی را مشخص می‌کند.

1	0	-1
1	0	-1
1	0	-1

7	7	0	0	6	7	0	0
7	7	0	0	6	7	0	0
7	3	0	0	7	7	0	0
7	7	0	0	7	0	0	0
7	7	0	0	7	0	0	0
0	7	0	0	7	0	0	0
3	1	0	0	7	0	0	0
3	0	0	0	7	0	0	0

(c) این فیلتر مرزها را از جنوب شرقی به شمال غربی مشخص می‌کند.

1	1	0
1	0	-1
0	-1	-1

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

(d) این فیلتر مشابه فیلتر a است با این تفاوت که همسایگی قطری را در نظر نمی‌گیرد.

0	-1	0
-1	4	-1
0	-1	0

0	7	7	0	0	6	0	6
0	7	0	0	0	0	7	0
0	7	2	6	6	0	0	6

0	7	0	0	0	7	0	0
0	0	7	0	6	0	6	0
5	0	7	0	3	0	3	0
7	0	7	0	6	0	3	2
2	7	0	0	0	7	0	7

(e) این فیلتر، نقاط تیرهای دورتر از محدوده‌های روشن هستند را مشخص می‌کنند. به بیان دیگر، مجموعه نقاط تیره متراکم را حفظ و سایر نقاط را روشن می‌کند.

1	1	1
1	1	1
1	1	1

7	7	7	7	6	7	7	7
7	7	7	7	7	7	7	7
7	7	7	7	7	7	7	7
7	7	7	7	7	7	7	7
7	7	7	7	7	7	7	6
7	7	7	7	7	7	7	7
7	7	7	7	7	7	7	7
7	7	7	7	7	7	7	7

(f) این فیلتر مرزهای شمال جنوبی را مشخص می‌کند.

-1	-1	-1
2	2	2
-1	-1	-1

7	7	7	7	3	2	5	2
0	0	0	0	0	5	7	7
0	6	7	7	7	1	0	2
7	0	0	0	0	0	0	0
0	0	0	7	0	2	0	3
0	0	0	0	0	0	0	0
5	7	4	7	0	2	0	2
5	0	0	0	7	5	7	7

(g) این فیلتر مشابه فیلتر C است و مرزهای جنوب غربی به شمال شرقی را مشخص می‌کند.

0	1	1
-1	0	1
-1	-1	0

0	0	7	7	0	0	7	7
0	0	7	6	0	0	0	7
0	0	7	0	0	0	0	0
0	0	7	1	0	0	7	0
0	0	7	7	0	0	7	6
7	0	0	7	0	0	6	7
0	0	0	1	0	1	4	7
0	0	0	0	0	0	0	0

(h) این فیلتر مشابه فیلتر f است و مرزهای غربی شرقی را مشخص می‌کند.

-1	2	-1
-1	2	-1
-1	2	-1

0	7	0	0	0	2	7	2
0	7	0	0	0	2	7	5
0	7	0	0	0	7	1	2
0	7	3	0	0	7	0	0
0	0	7	0	0	7	0	0
7	0	7	0	7	0	6	0
7	0	7	0	0	7	0	7
7	0	7	0	0	7	0	7

(i) رویکرد اصلی برای حل سوالات این قسمت حدس زدن می‌باشد. البته برای پیاده سازی الگوریتم می‌توان از ایجاد دستگاه معادلات به ویژه در نقاط مرزی استفاده کرد.

ماتریس خروجی در این قسمت دارای اعداد گرد شده می‌باشد بنابراین ضرایب فیلتر شامل اعداد اعشاری می‌باشند. با توجه به بلاک‌های مشخص شده در شکل زیر، فیلتر این قسمت یک فیلتر میانگین با ضرایب برابر بوده است.

0	7	7	0	0	3	3	3
0	7	0	0	0	3	7	3
0	7	3	3	3	3	3	3
0	7	0	0	0	7	0	0
0	0	7	0	3	3	3	0
3	0	7	0	3	3	3	0
7	0	7	0	3	3	3	3
3	3	0	0	0	7	0	7

شکل ۱-۲

(j) با توجه به خروجی‌ها این فیلتر احتمالاً فاقد اعداد اعشاری می‌باشد. با بررسی نقاطی که در آن‌ها خروجی صفر است و همچنین همسایه‌های آن می‌توان فهمید که فیلتر این قسمت به شکل زیر است.

1	1	1
---	---	---

0	0	0
1	1	1

ک) این فیلتر نیز دارای اعداد اعشاری است. با توجه به بلاک‌های مشخص شده در شکل ۲-۲، به راحتی می‌توان فهمید این فیلتر به شکل زیر است.(اعداد کسری هستند)

0	1/3	0
0	1/3	0
0	1/3	0

0	7	7	0	0	3	3	3
0	7	0	0	0	3	7	3
0	7	3	3	3	3	3	3
0	7	0	0	0	7	0	0
0	0	7	0	3	3	3	0
3	0	7	0	3	3	3	0
7	0	7	0	3	3	3	3
3	3	0	0	0	7	0	7

شکل ۲-۲

ا) این فیلتر دارای اعداد منفی و فاقد اعداد اعشاری است. مجدداً با توجه به بلاک‌های مشخص شده در شکل ۲-۲ می‌توان فهمید این فیلتر به شکل زیر است.

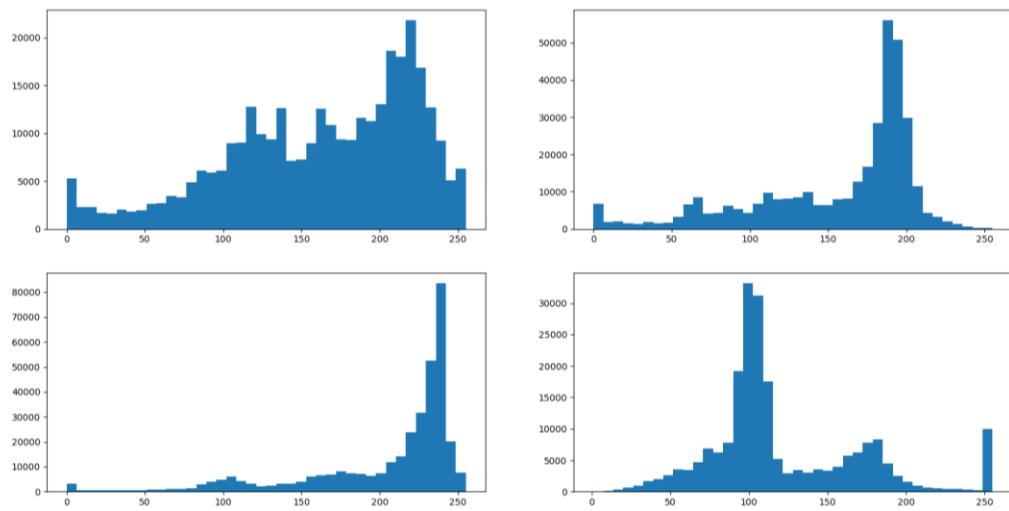
0	-1	0
-1	5	-1
0	-1	0

(۳)

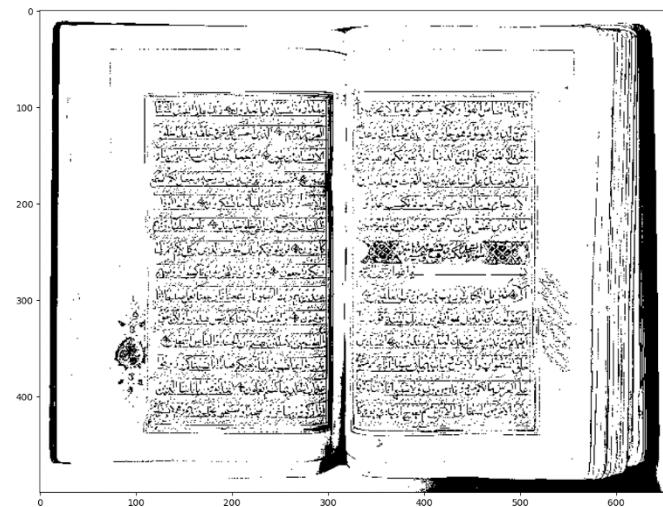
ب) کد مربوط به این قسمت در فایل `a.py` قرار دارد. در شکل ۱-۳ و ۲-۳ تصاویر سطح خاکستری و هیستوگرام مربوط به آن‌ها نمایش داده شده است. انتخاب حد آستانه با توجه به این هیستوگرام‌ها و سعی و خطا بوده است. در شکل‌های ۳-۳ تا ۶-۳ تصاویر خروجی و حد آستانه اعمال شده نمایش داده شده است. انتخاب حد آستانه به گونه‌ای بوده که سایه‌های موجود در تصویر تا حد ممکن حذف شوند. این کار همانطور که مشاهده می‌شود در بعضی ناحیه‌ها موجب ناخوانا شدن متن کتاب شده است.



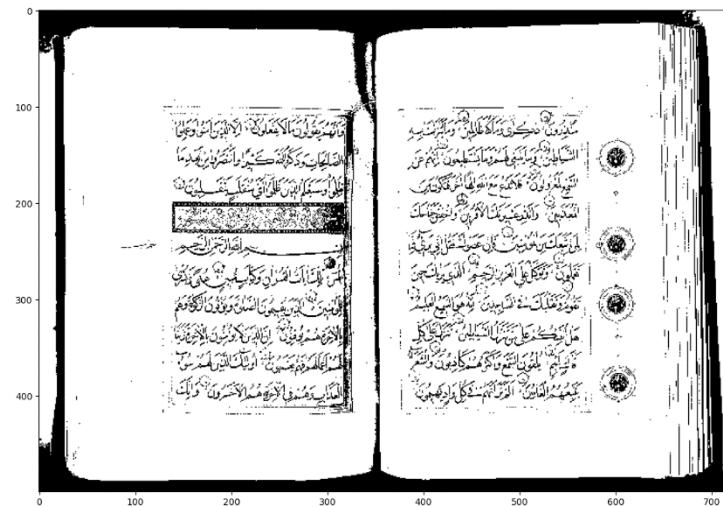
شکل ۱-۳



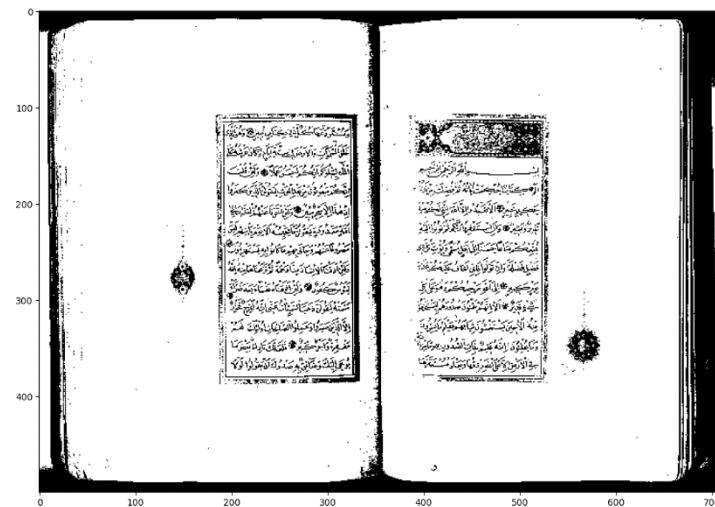
شکل ۲-۳



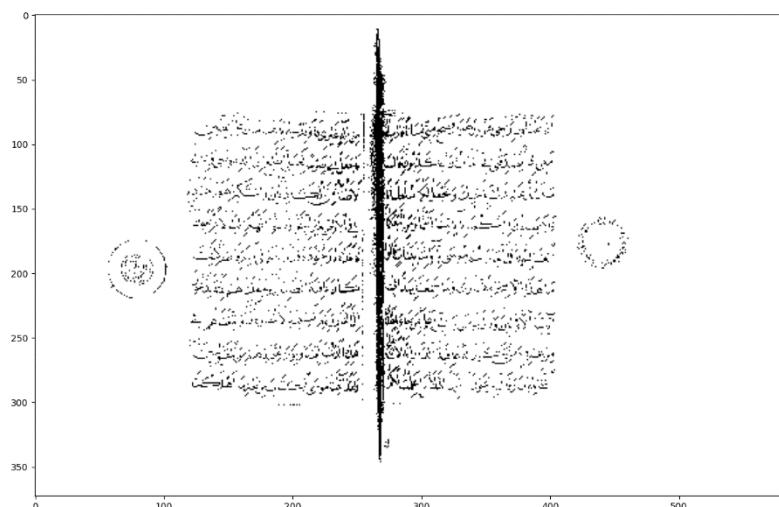
شکل ۳-۳- حد آستانه ۹۵



شکل ۳-۴- حد آستانه ۱۰۰

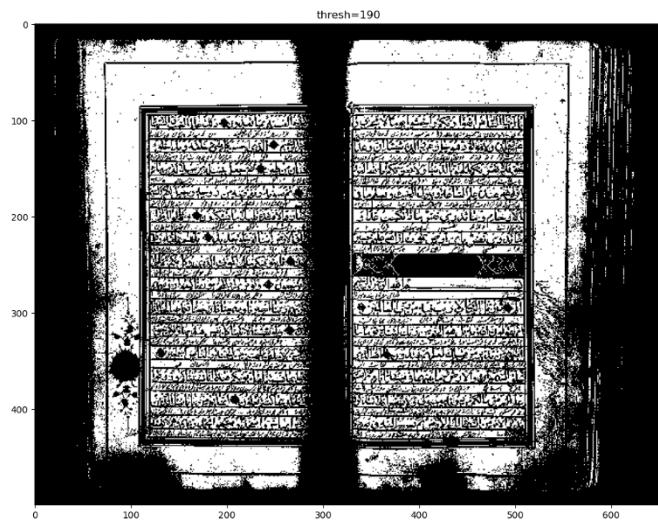


شکل ۳-۵- حد آستانه ۵۶

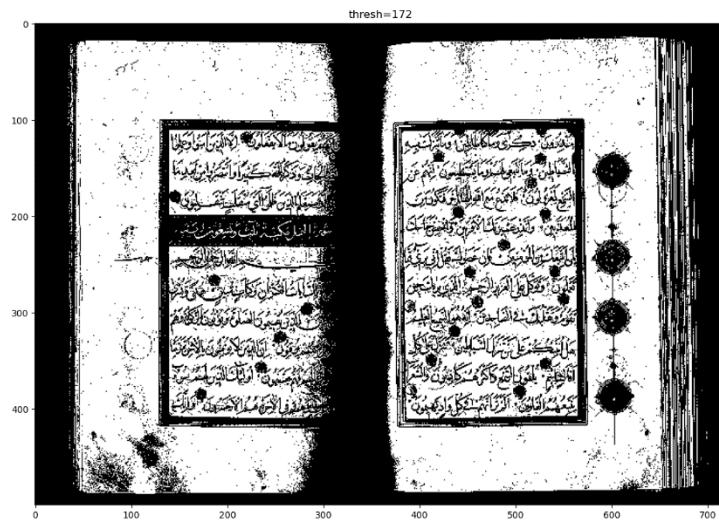


شکل ۳-۶- حد آستانه ۵۰

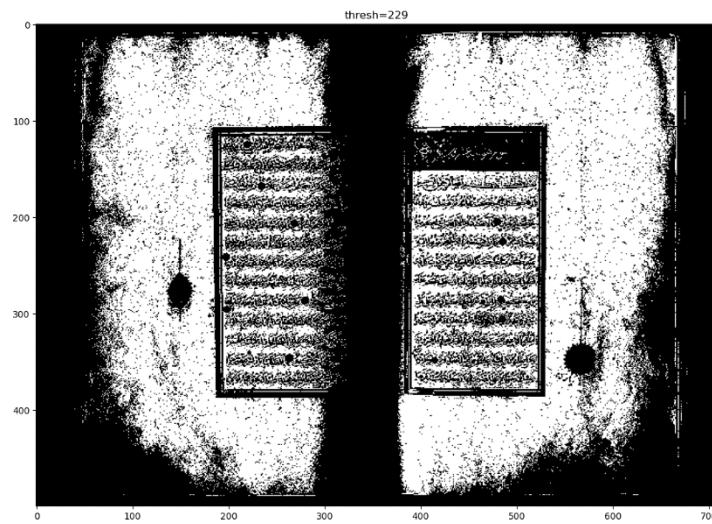
(b) کد مربوط به این قسمت در فایل `b.py` قرار دارد. خروجی‌ها و حد آستانه انتخاب شده توسط الگوریتم در شکل‌های ۷-۳ تا ۱۰-۳ ارائه شده است.
همانطور که مشاهده می‌شود حد آستانه انتخاب شده برای هیچکدام از تصاویر مناسب نیست.



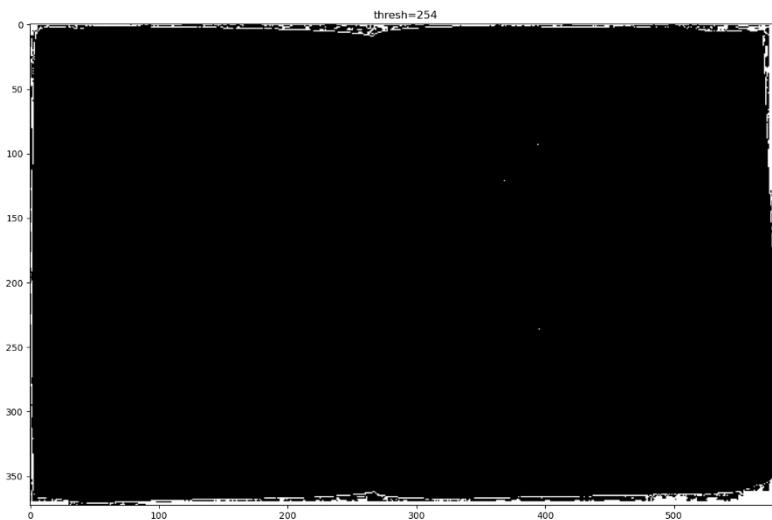
شكل ٢-٣



شكل ١-٣



شکل ۹-۳

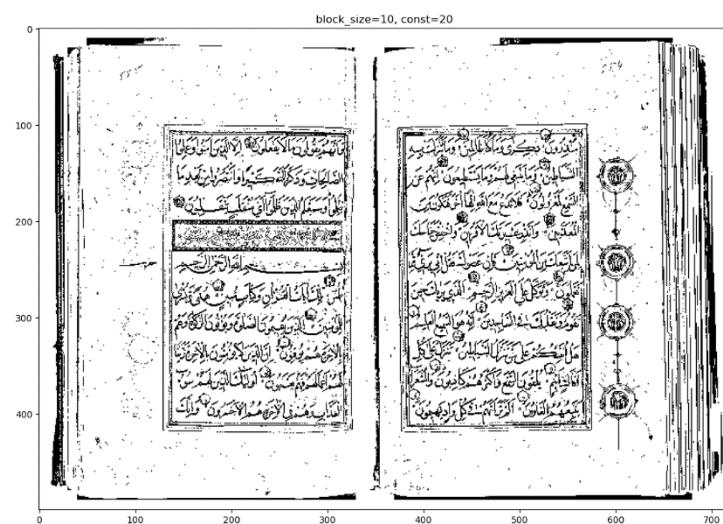


شکل ۱۰-۳

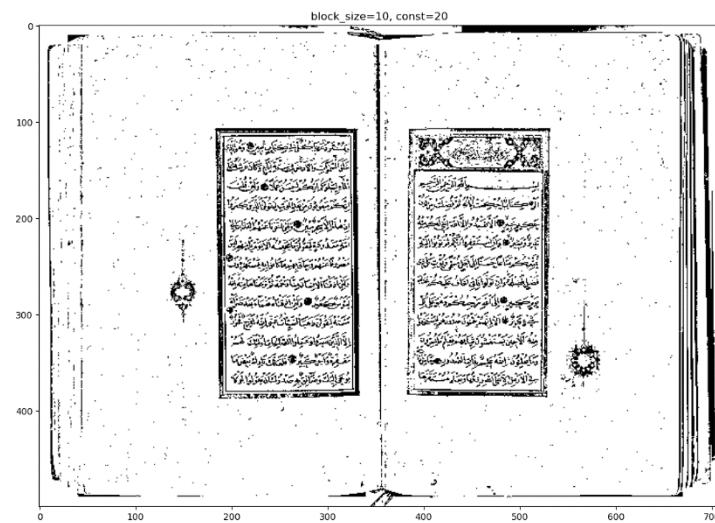
۶) کد مربوط به این بخش در فایل C.py قرار دارد. انتخاب پارامترها به گونه‌ای بوده است که ضمن حفظ ساختار صفحه، سایه‌های موجود در آن از بین بروند. تصاویر خروجی و پارامترهای آن‌ها در شکل‌های ۱۱-۳ تا ۱۴-۳ نمایش داده شده‌است.



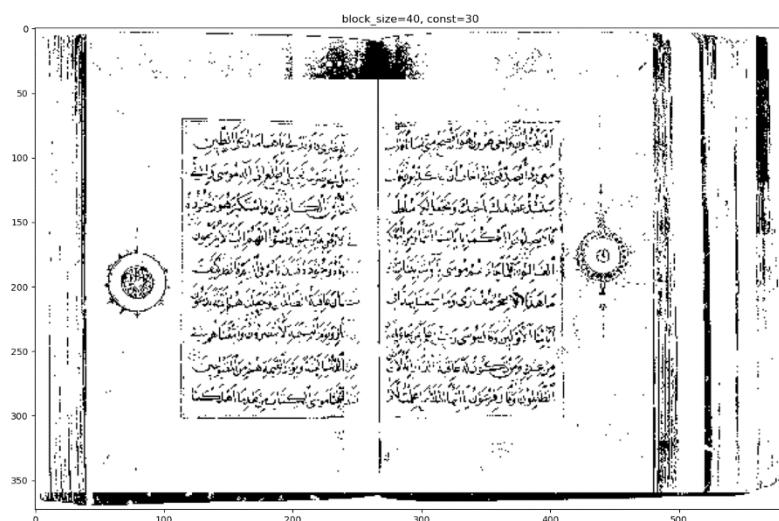
شكل ١١-٣



شكل ١٢-٣



شکل ۱۳-۳

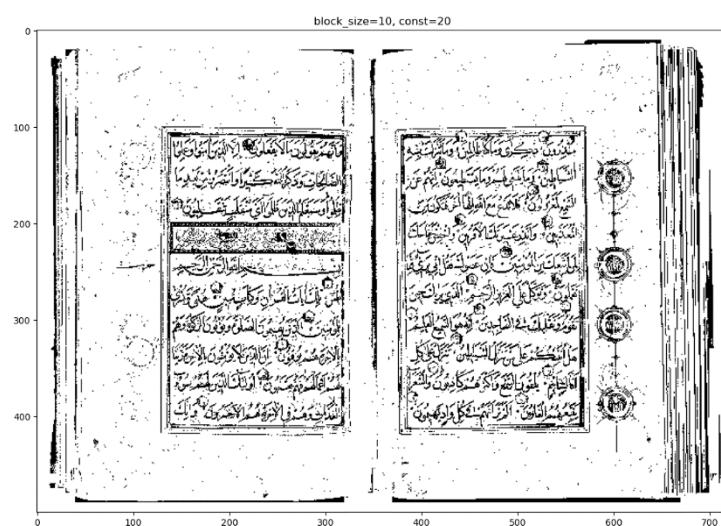


شکل ۱۴-۳

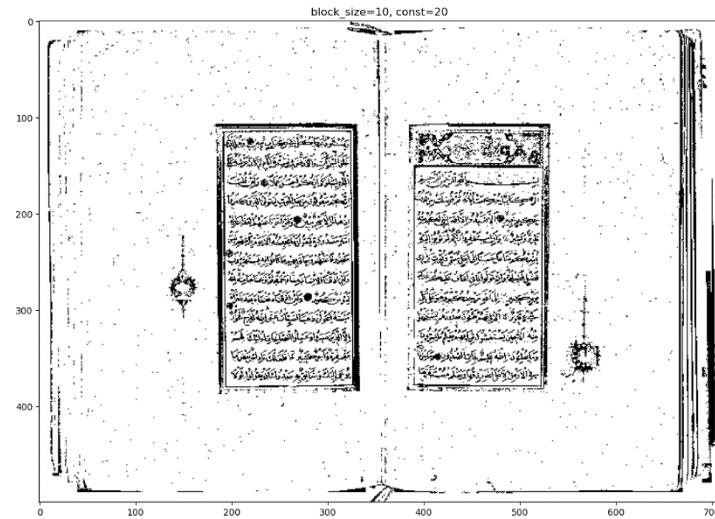
(d) کد مربوط به این بخش در فایل `d.py` قرار دارد. تصاویر خروجی و پارامترهای آن‌ها در شکل‌های ۱۵-۳ تا ۱۸-۳ نمایش داده شده است.



شكل ٣



شكل ٤



شکل ۱۷-۳

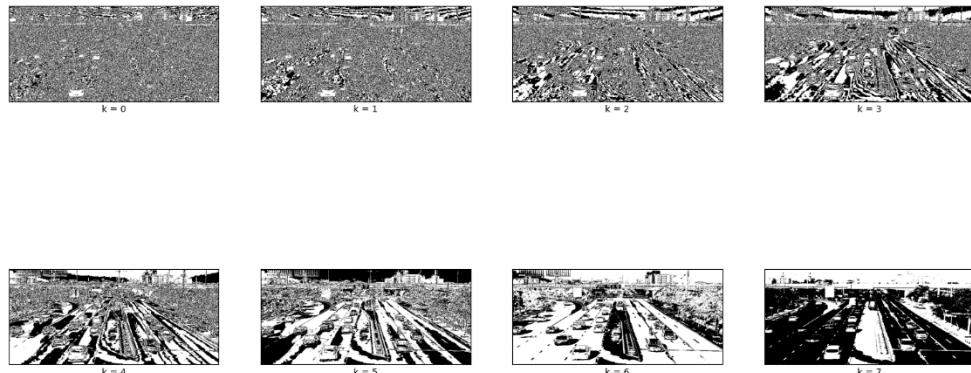


شکل ۱۸-۳

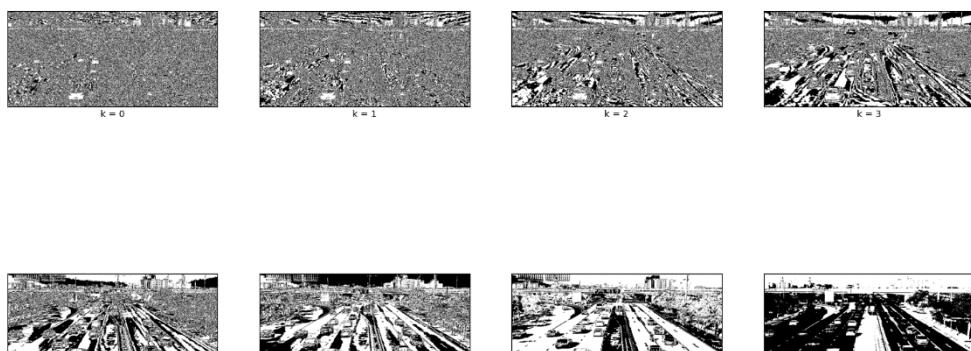
(e) روش‌های global thresholding و Otsu با توجه به اینکه ترشیه‌ولدینگ سراسری بر روی تصاویر اعمال می‌کند نتایج چندان مطلوبی ندارند. در روش global thresholding از آنجا که انتخاب حد آستانه با سعی و خطا و به صورت دستی تعیین شده‌است، خروجی بهتری حاصل شده است. اگرچه در این روش نیز بخشی از متن‌های موجود در تصویر ناخوانا شده است. همچنین انتخاب حد آستانه در روش Otsu بر اساس حداکثر کردن واریانس بین دو طرف حد آستانه است. این روش با توجه به وجود سایه در بخش زیادی از تصاویر در اینجا بدترین عملکرد را داشته است.

هر دو روش انطباقی بر روی این تصاویر عملکرد نسبتاً مطلوبی داشته‌اند. روش انطباقی گاووسی در بلاک‌های نزدیک به ناحیه‌های سایه، به دلیل اهمیت دادن بیشتر به پیکسل‌های مرکزی، اندکی بهتر عمل کرده است. ضعف کوچکی که هر دو روش انطباقی دارند، وجود نقاط سیاه پراکنده روی صفحات است. با تغییر در پارامترها می‌توان این نقاط را از بین برد اما این کار باعث کم شدن جزئیات صفحه و ناخوانا شدن بعضی بخش‌های متن می‌شود. بنابراین از حذف آن‌ها صرف نظر شده است. (همانطور که گفته شد انتخاب پارامترها به گونه‌ای بوده است که تا حد امکان فقط سایه از تصاویر حذف شود)

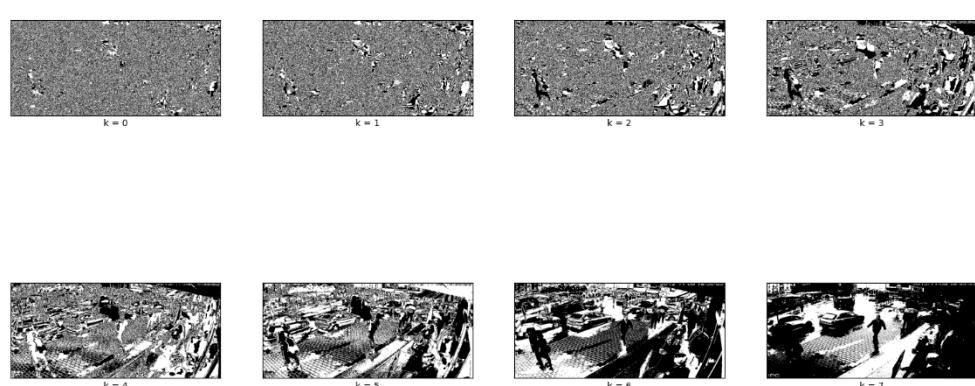
(a) کد مربوط به این قسمت در فایل `a.py` قرار دارد.تابع خواسته شده پس از پیداهسازی، برای هرچهار تصویر فراخوانی شده است. نتایج در شکل ۱-۴ تا ۴ ارائه شده است.



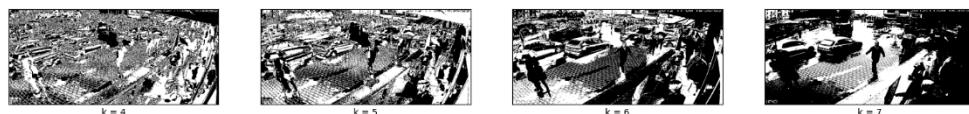
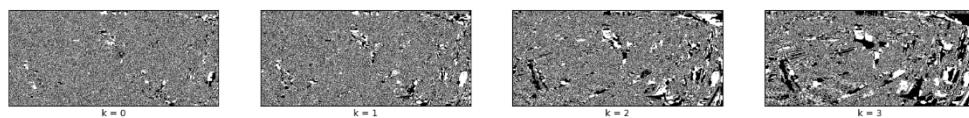
شکل ۱-۴



شکل ۲-۴

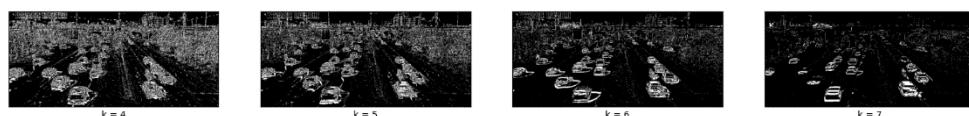
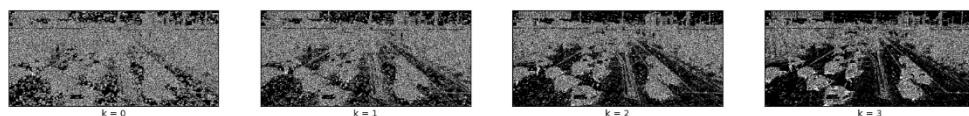


شکل ۳-۴

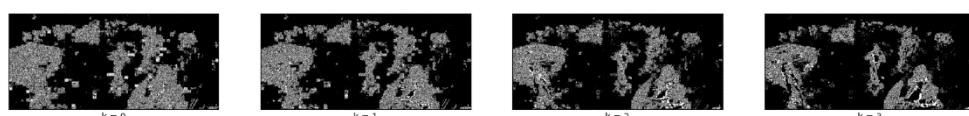


شکل ۴-۵

(b) کد مربوط به این قسمت در فایل `b.py` قرار دارد. خروجی‌ها برای هر جفت از عکس‌ها در شکل ۴-۵ و ۴-۶ نمایش داده شده است.



شکل ۴-۷



شکل ۴-۱۰

(c) کد مربوط به این قسمت در فایل `c.py` قرار دارد. خروجی‌ها در شکل ۴-۷ و ۴-۸ ارائه شده است.

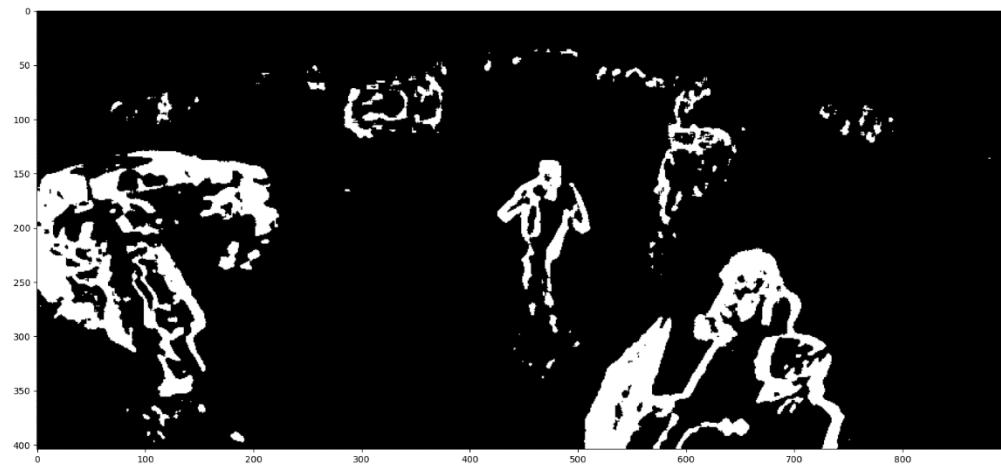


شکل ۷-۴

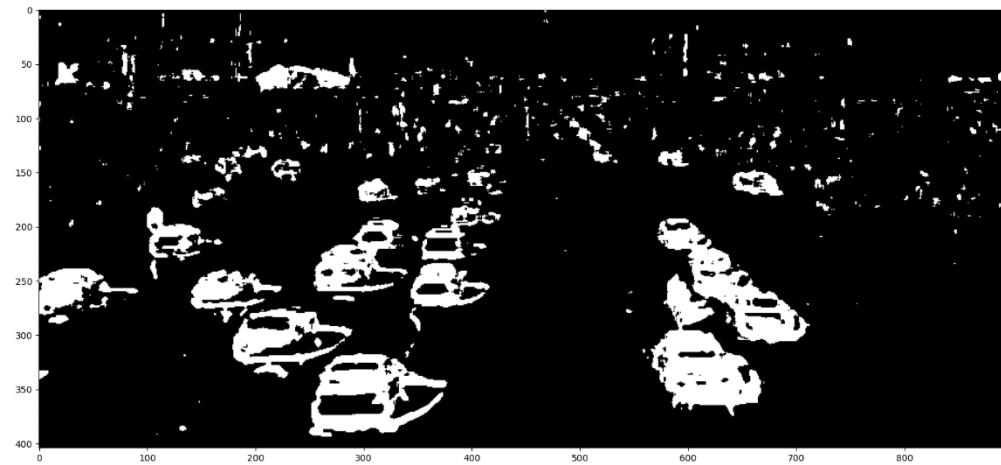


شکل ۸-۴

(d) کد مربوط به این قسمت در فایل `d.py` قرار دارد. برای بهبود خروجی‌های قسمت C، از یک فیلتر میانه 5×5 و سپس ترشحولدینگ با مقدار ۴۰ انجام شده است. خروجی‌ها در تصاویر ۹-۴ و ۱۰-۴ ارائه شده‌است. همانطور که مشاهده می‌شود، برخی اجسام متحرک که در فاصله دورتری از دوربین قرار دارند، به علت جایه‌جایی کمتر نسبت به دوربین، به همراه نویزها حذف شده‌اند.



شکل ۴-۳



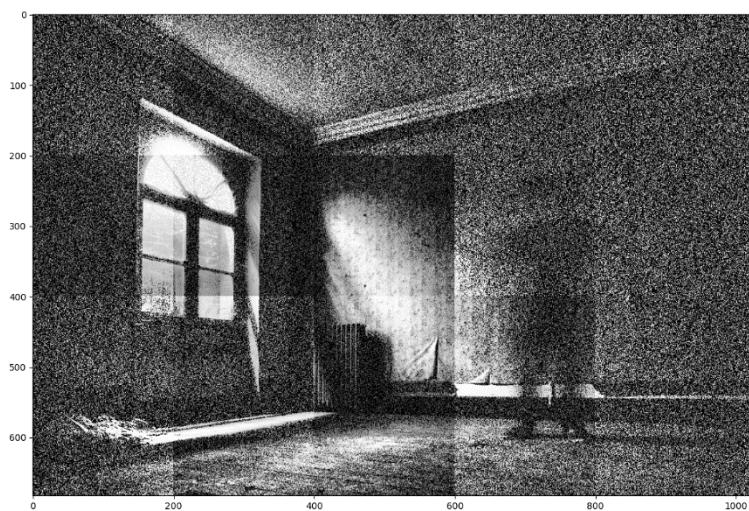
شکل ۵-۳

(۵)

a) کد مربوط به این قسمت در فایل `a.py` قرار دارد. با فرض ابعاد 200×200 برای بلاک‌ها، خروجی‌ها در شکل‌های ۱-۵ تا ۴-۵ نمایش داده شده است.



شكل ١



شكل ٢



شکل ۳-۵



شکل ۴-۵

(b) کد مربوط به این قسمت در فایل `b.py` قرار دارد. بر روی نواحی مرزی، فیلتر smoothing اعمال می‌کنیم. نتایج در شکل‌های ۵-۵ تا ۸-۵ ارائه شده است. همانطور که مشاهده می‌شود انجام این کار تنها باعث اندکی بهبود در نواحی مرزی شده است.



شكل ٥-٥



شكل ٥-٦



شکل ۷-۵



شکل ۸-۵

(۶)

برای این تمرین خروجی‌ها برای هر دو حالت میانگین و میانه باهم نمایش داده می‌شود. همچنین خروجی‌ها به تفکیک تعداد فریم و متند استفاده شده (میانگین یا میانه) در پوشش‌های مربوط به هر بخش ذخیره شده‌اند.

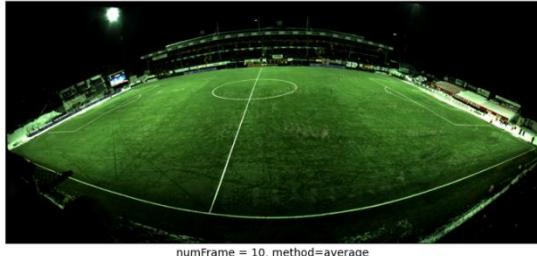
(a) کد مربوط به این قسمت در فایل `a.py` قرار دارد. خروجی‌ها برای میانگین و میانه در شکل‌های ۱-۶ و ۲-۶ نمایش داده شده‌است. همانطور که مشاهده می‌شود متند میانه در این تصاویر نتایج بهتری را ایجاد کرده‌است. دلیل این اختلاف این است که این متند، بیشترین مقدار تکرار شده را جایگزین می‌کند و در این حالت سایر مقادیر در تعیین رنگ پیکسل، نقشی ندارند.



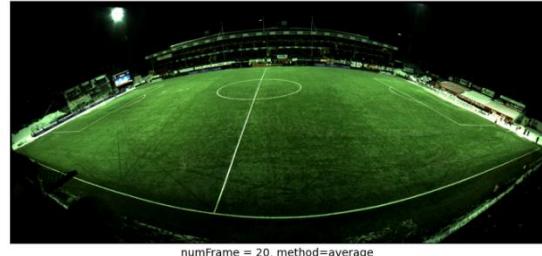
numFrame = 2, method=average



numFrame = 5, method=average



numFrame = 10, method=average



numFrame = 20, method=average

شکل ۶-۱



numFrame = 2, method=median



numFrame = 5, method=median



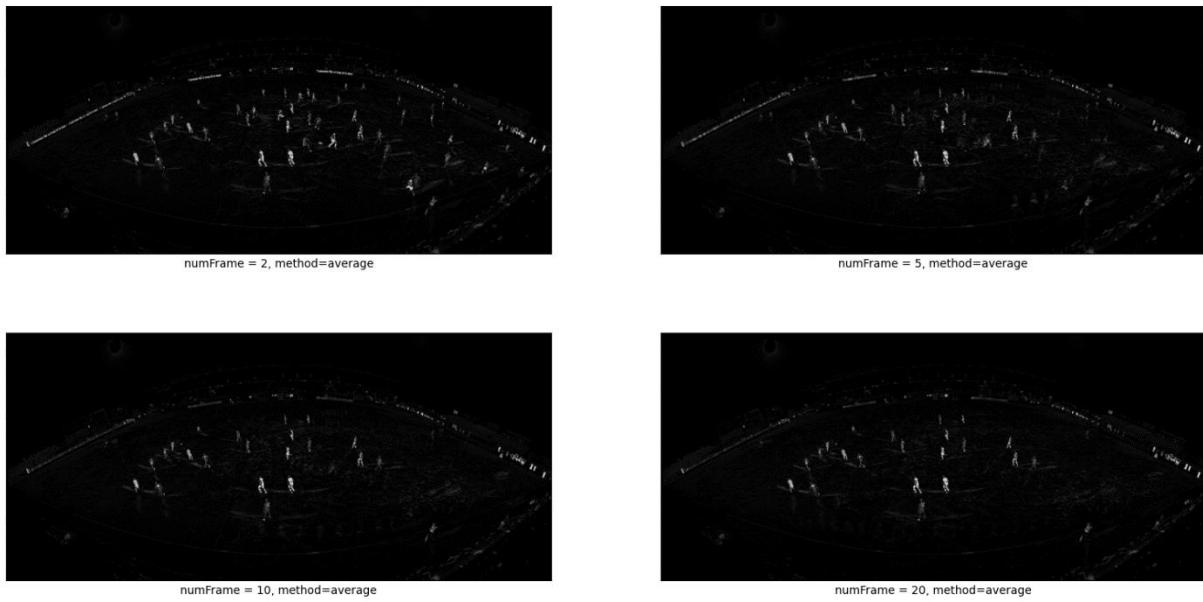
numFrame = 10, method=median



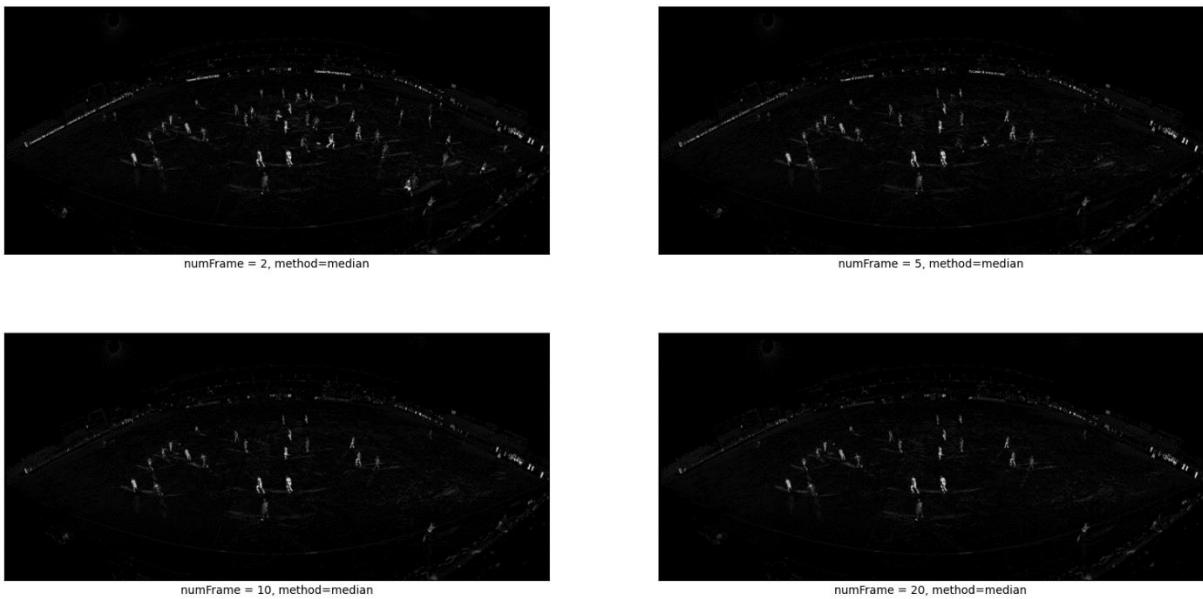
numFrame = 20, method=median

شکل ۶-۲

(b) کد مربوط به این قسمت در فایل **b.py** قرار دارد. قدر مطلق اختلاف دو تصویر حساب شده و به صورت سطح خاکستری ذخیره شده است. نتایج در تصاویر ۳-۶ و ۴-۶ نمایش داده شده است.



شکل ۶-۳



شکل ۶-۴

۶) کد مربوط به این قسمت در فایل C.py قرار دارد. برای تصاویر ۲، ۵، ۱۰ و ۲۰ فریم، ترشهولیدینگ با حد به ترتیب ۷۰، ۴۰، ۴۰ و ۴۰ اعمال شده است. نتایج در شکل های ۶-۵ و ۶-۶ نمایش داده شده است. همانطور که مشاهده می شود، شبیه های موجود در تصاویر ۲ و ۵ فریم، به خصوص برای متد میانگین، در ماسک ها ظاهر شده اند. تغییر حد ترشهولیدینگ برای حذف این بخش ها باعث کاهش جزئیات ماسک واقعی خواهد شد.



numFrame = 2, trh = 70, method=average



numFrame = 5, trh = 60, method=average



numFrame = 10, trh = 40, method=average



numFrame = 20, trh = 40, method=average

شکل ۵-۵



numFrame = 2, trh = 70, method=median



numFrame = 5, trh = 60, method=median



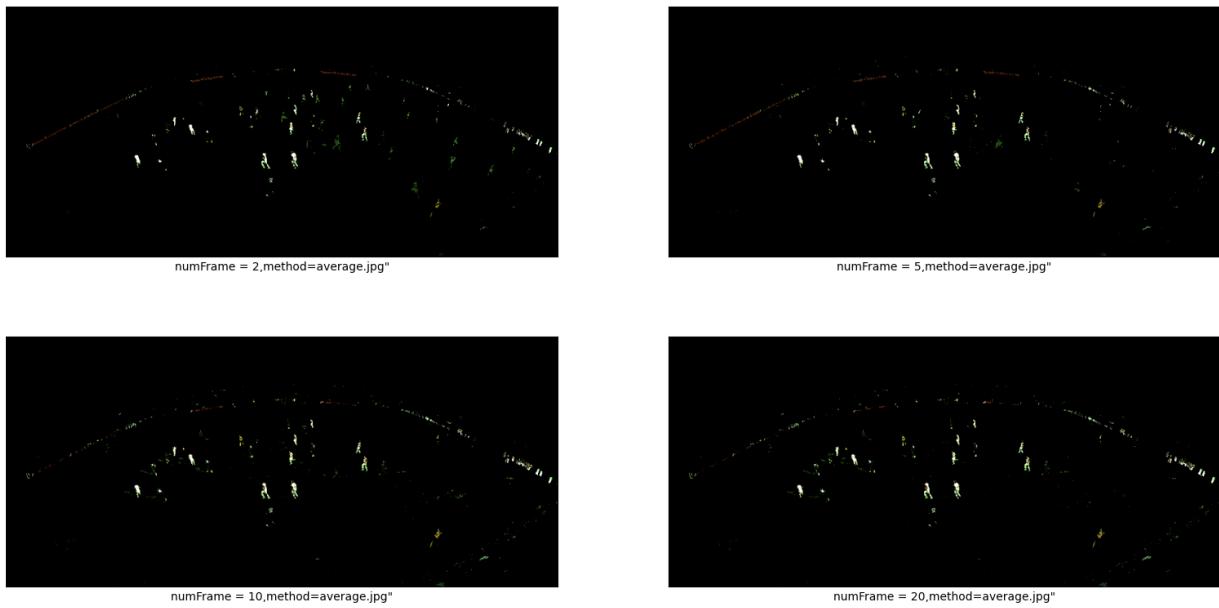
numFrame = 10, trh = 40, method=median



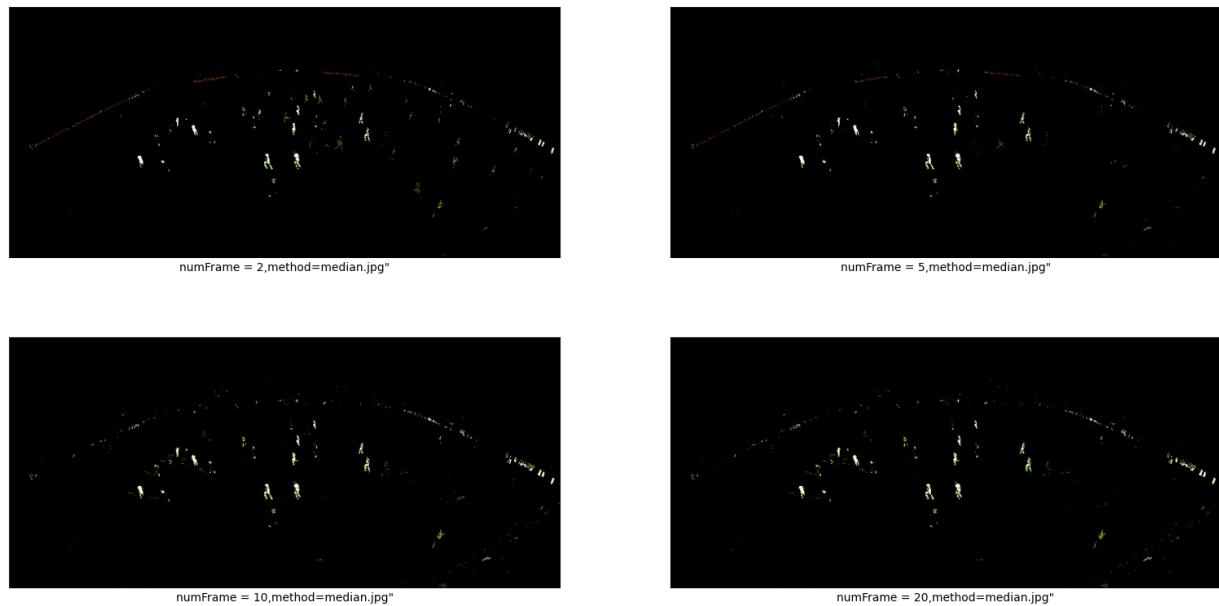
numFrame = 20, trh = 40, method=median

شکل ۵-۶

(d) کد مربوط به این بخش در فایل **d.py** قرار دارد. نتایج در شکل های ۶-۷ و ۸-۹ ارائه شده است. با توجه به اینکه لباس یکی از تیمها مشکی رنگ است، بازیکنان این تیم به خوبی در تصویر خروجی قابل مشاهده نیستند. همچنین وجود شیخ در تصاویر ۲ و ۵ فریم، کیفیت خروجی را تحت تاثیر قرار داده است.



شکل ۶



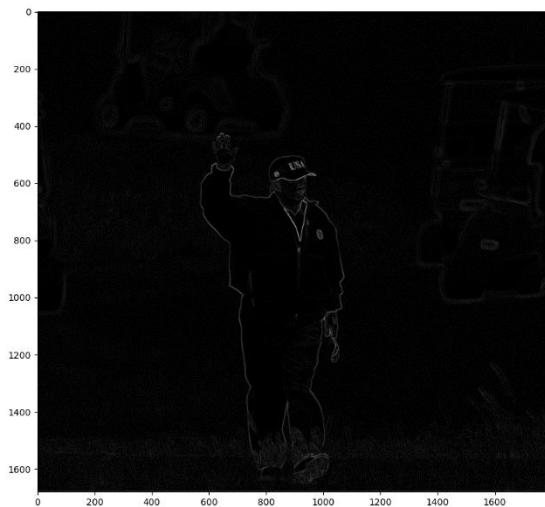
شکل ۷

(۷)

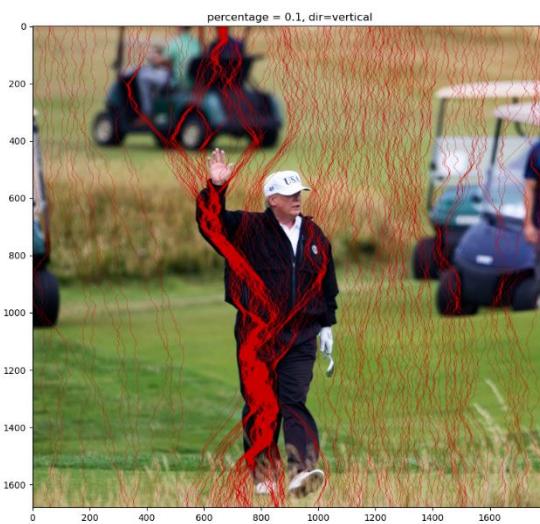
برای همه قسمت‌های این سوال از یک کد استفاده می‌شود. این کد در فایل `image-comp.py` قرار دارد. در خطوط ۹۷ تا 10^4 می‌توان پارامترهای ورودی از جمله آدرس عکس، درصدهای کاهش سایز و جهت کاهش سایز را تعیین کرد.

تابع `find_seams_list` با استفاده از یک حلقه شروع به پیدا کردن `seem` ها می‌کند. در هر مرحله از حلقه، یک `seem` جدید یافت می‌شود، مقادیر آن به فضای تصویر اصلی منتقل و سپس آن `seem` از تصویر حذف می‌شود. لازم به تأکید است `seem` های پیدا شده به فضای تصویر اصلی (کاهش نیافنه) منتقل می‌شوند؛ بنابراین خروجی این تابع دقیقاً مطابق خواسته سوال است. همچنین یک ماتریس با نام `index_map` که هر پیکسل را به یک `seem` نگاشت می‌کند، از این تابع دریافت می‌شود که برای رسم `seem` ها در مراحل بعدی از آن استفاده می‌شود.

(a) خروجی‌ها در شکل‌های ۱-۷ تا ۱۴-۷ نمایش داده شده‌است. همانطور که مشاهده می‌شود، `seem` های یافت شده در هر دو تصویر مناسب نیستند و در نتیجه تصویر خروجی مطلوب نیست. دلیل این اتفاق احتمالاً این است که لباس سوژه در هر دو تصویر گرادیان پایینی دارد و درنتیجه هزینه `seem` های عبوری از آن کمتر است.



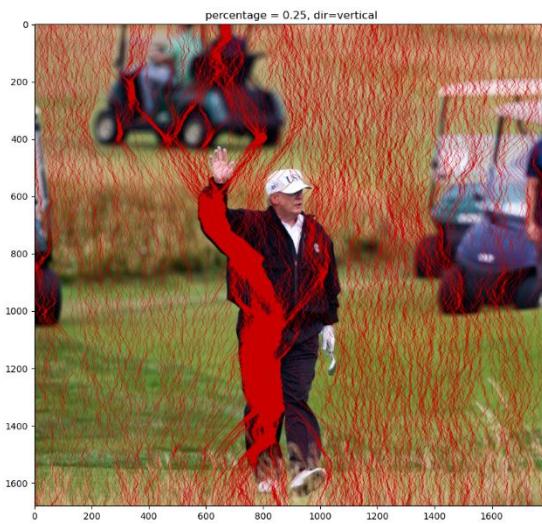
شکل ۱-۷



شکل ۲-۷



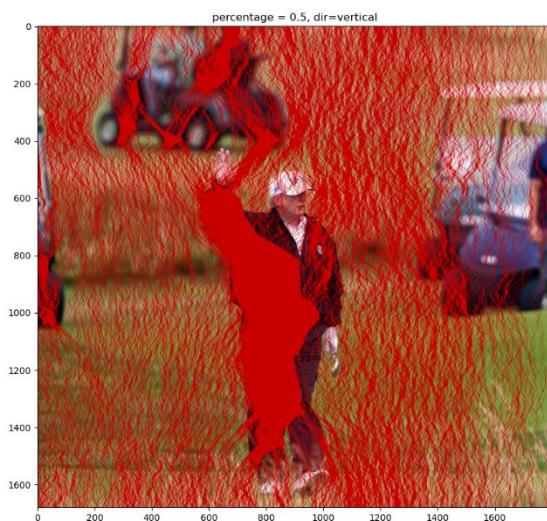
مکمل ۷



مکمل ۷



شكل ٦



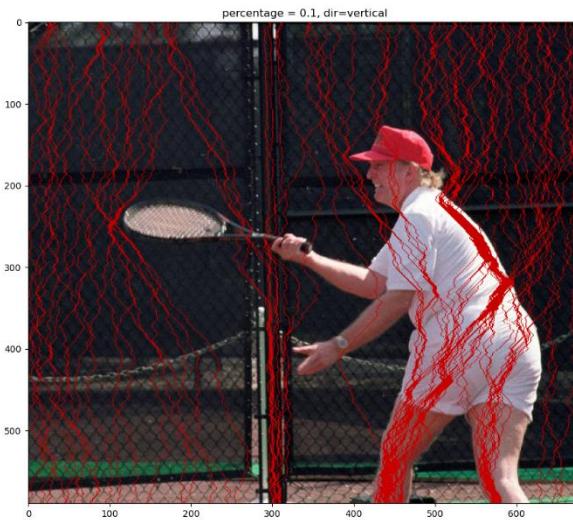
شكل ٧



شکل ۷-۷



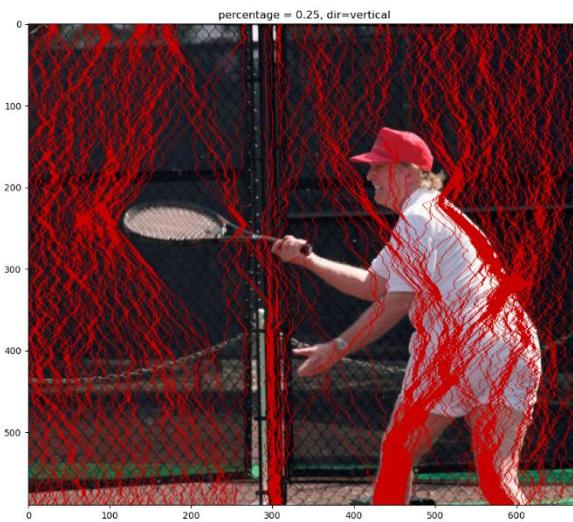
شکل ۸-۷



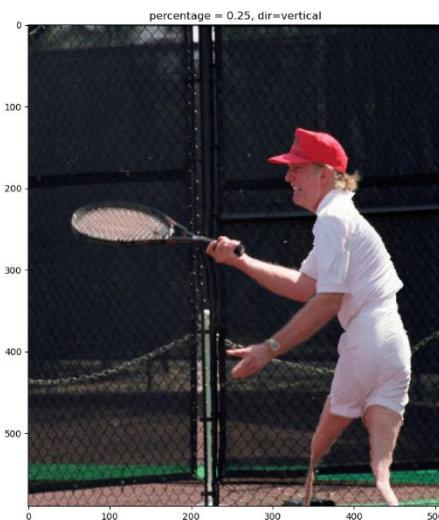
شكل ٧



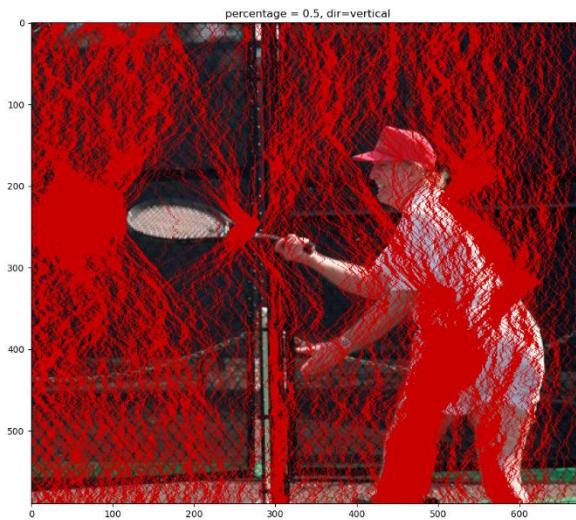
شكل ٨



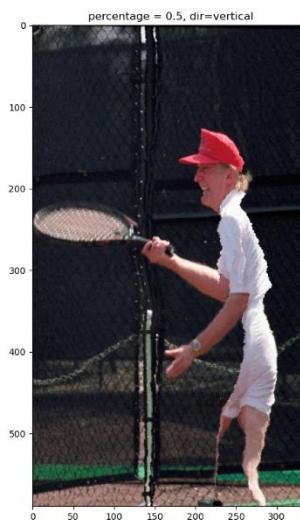
شكل ١١-٧



شكل ١٢-٧



شکل ۱۳-۷



شکل ۱۴-۷

(b) خروجی در شکل‌های ۱۵-۷ تا ۲۸-۷ نمایش داده شده است. در این بخش نیز خروجی به جز برای حالت ۱۰ درصد کاهش، کیفیت مناسبی ندارد. دلیل این اتفاق مشابه دلیلی است که برای بخش a توضیح داده شد.



شکل ۷-۱۵



شکل ۷-۱۶



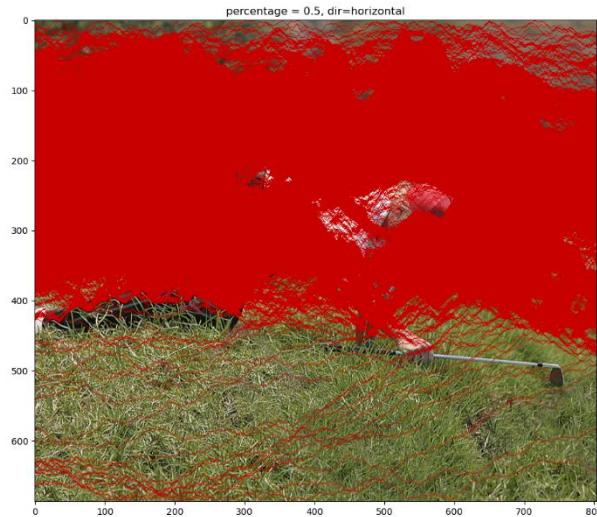
شكل ١٧-٧



شكل ١٨-٧



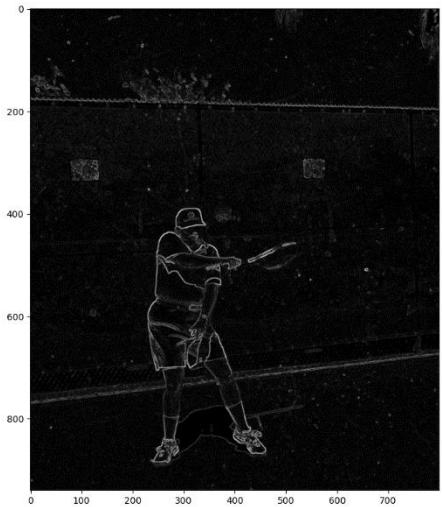
شكل ١٩-٧



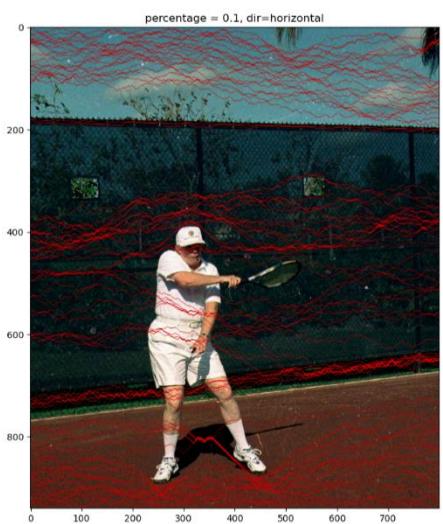
شكل ٢٠-٧



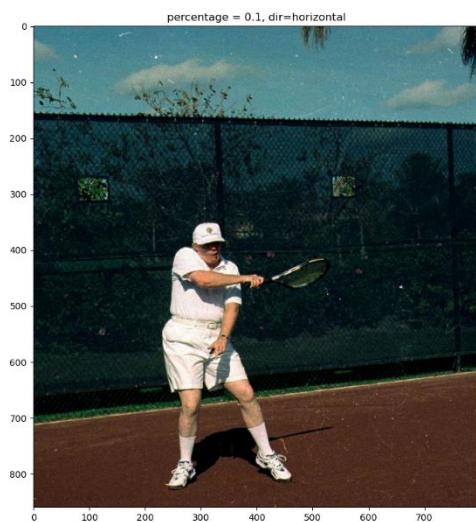
شكل ٢١-٧



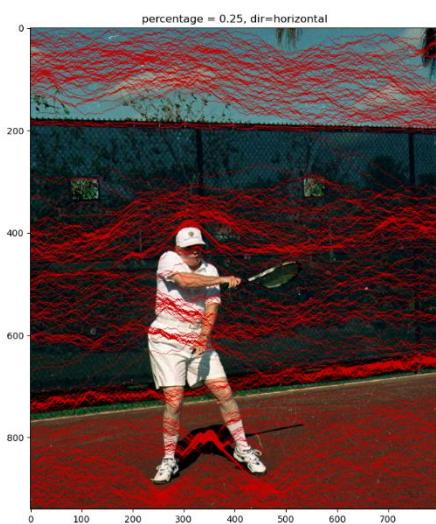
شکل ۷-۲



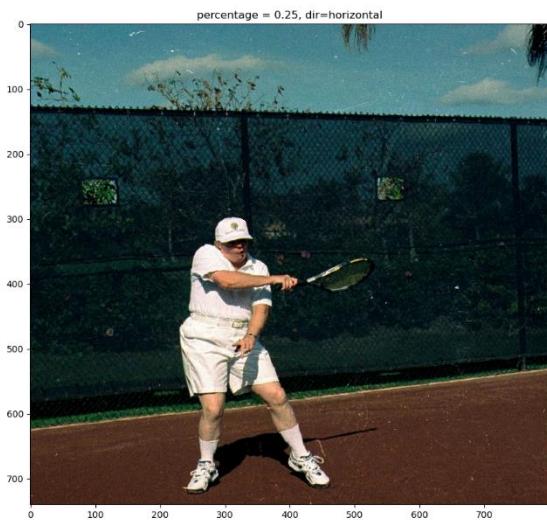
شکل ۷-۳



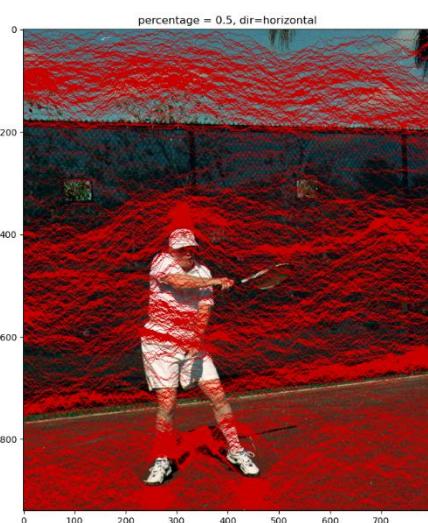
مکمل ۷



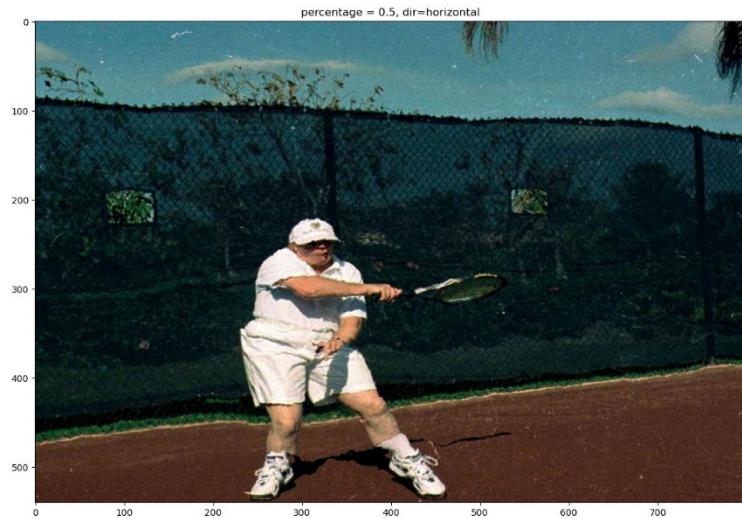
مکمل ۷



شکل ۷



شکل ۷



شکل ۷

۳) برای این قسمت تصویری که در شکل ۲۹-۷ مشاهده می‌شود انتخاب شده است. این تصویر همچنین در پوشه P7 موجود است. در شکل ۳۰-۷ و ۳۱-۷ نتایج این تصویر ارائه شده است.



شکل ۷



شکل ۳۱-۷



شکل ۳۱-۷

(d) برای این قسمت تصویری که در شکل ۳۲-۷ مشاهده می‌شود انتخاب شده است. این تصویر همچنین در پوشه P7 موجود است. در شکل ۳۳-۷ و ۳۴-۷، نتایج این تصویر ارائه شده است. برای این تصویر، به دلیل پایین بودن گرادیان سوزه‌ها، نتیجه مطلوبی بدست نیامده است.



شكل ٣٢-٧



شكل ٣٣-٧



شكل ٣٤-٧

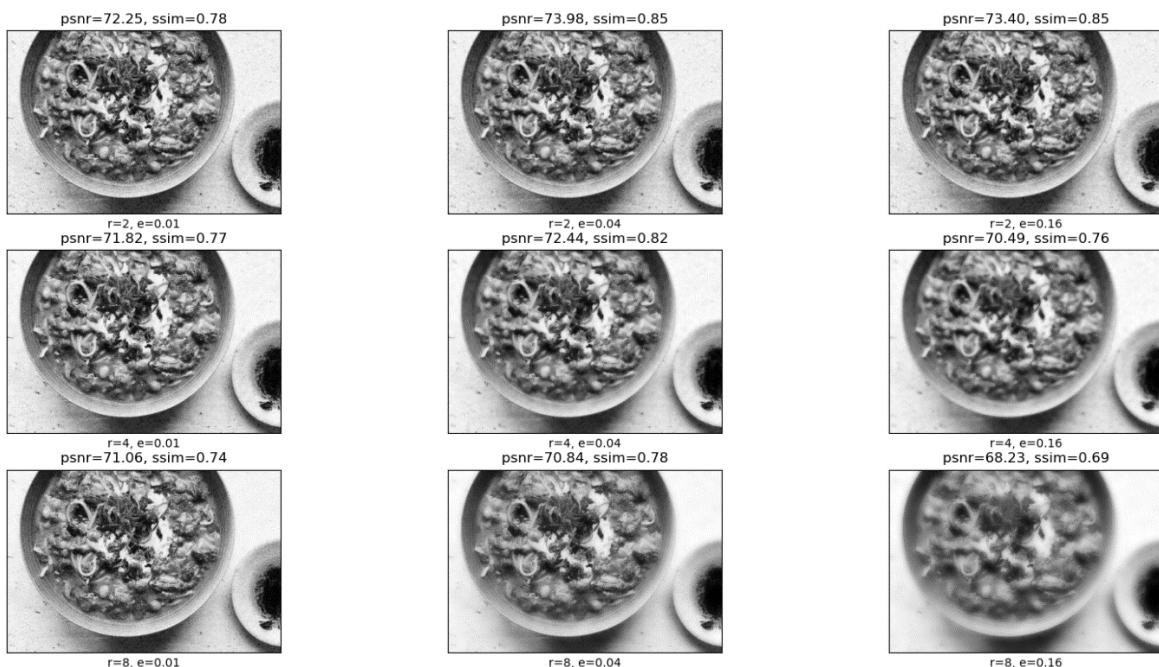
(۸)

نتایج مربوط به همه بخش‌های این سوال در پوشه‌های مجزا ذخیره شده‌اند.

(a) کد مربوط به این بخش در فایل `a.py` قرار دارد. تصویری نویزی در شکل ۱-۸ و نتایج به ازای پارامترهای مختلف در شکل ۲-۸ ارائه شده‌است.



شکل ۱-۸



شکل ۲-۸

(b) از نظر بصری، افزایش هر دو پارامتر باعث افزایش افکت بلور در تصویر می‌شود. به لحاظ پارامترهای ارزیابی، افزایش ۲ باعث کاهش کیفیت تصویر و افزایش e ابتدا باعث افزایش و سپس باعث کاهش آن می‌شود.

(c) کد مربوط به این بخش در فایل `c.py` قرار دارد. در تصویر `3-3-alf` تصویر فیلتر شده با تصویر نویزی و در تصویر `3-3-B` تصویر فیلتر شده با تصویر اصلی نمایش داده شده است. مقدار پارامتر `z` برابر با `4` و مقدار پارامتر `e` برابر با `0.01` می باشد. همانطور که مشاهده می شود استفاده از تصویر اصلی به عنوان تصویر راهنمای باعث کاهش بیشتر نویز در خروجی شده است؛ هرچند افکت بلور تصویر نیز بیشتر شده است و احتمالاً به همین دلیل معیارهای ارزیابی برای آن مقدار کمتری دارد. با تغییر پارامترها می توان این ایراد را نیز برطرف کرد.



شکل ۱-۳-ب - $ssim=0.83$ $psnr=72.76$

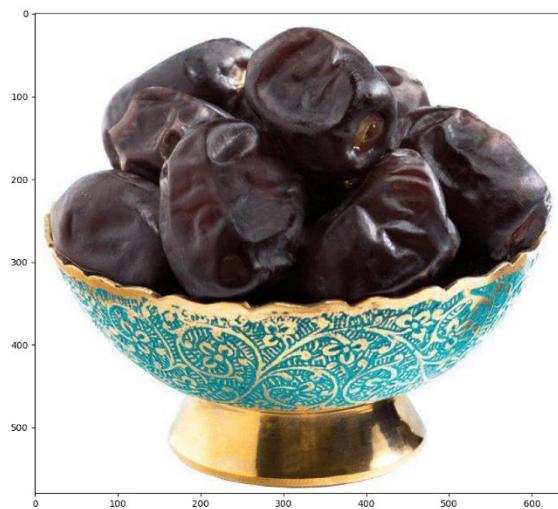
شکل ۱-۳-الف $ssim=0.85$ $psnr=73.98$

(d) کد مربوط به این قسمت در فایل `d.py` قرار دارد. پارامترهای مناسب با آزمون و خطای پیدا شده و نتیجه در شکل `4-8` ارائه شده است. همانطور که مشاهده می شود نتیجه فیلتر گاووسی از نظر کیفی در مقایسه با روش قبلی بسیار ضعیف است، هرچند از نظر کمی تفاوت خاصی با روش های قبلی ندارد.

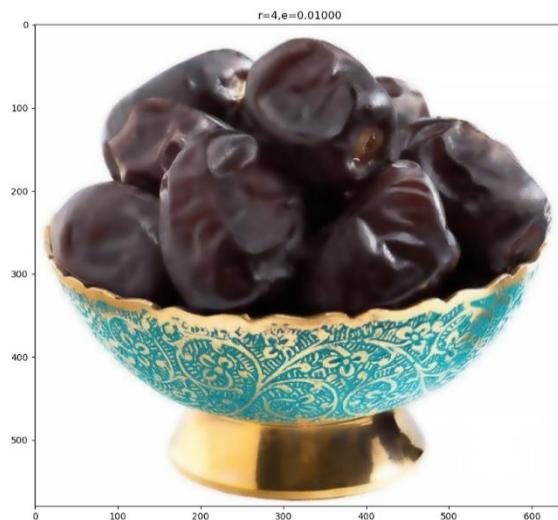


شکل ۴

(e) کد مربوط به این قسمت در فایل `e.py` قرار دارد. تصویر اصلی در شکل `5-8` و بهترین خروجی از نظر بصری در شکل `6-8` نشان داده شده است. در این قسمت و قسمت `f`، معیار اصلی برای انتخاب بهترین خروجی، تصویری بوده است که ضمن ایجاد افکت بلور، لبه ها را تا حد ممکن حفظ کند.

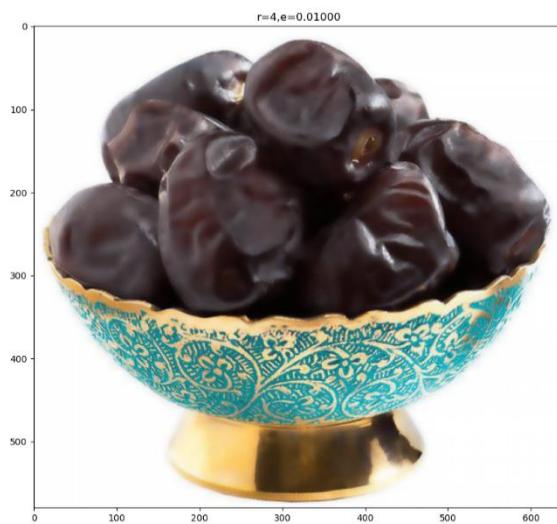


شکل ۵-۱



شکل ۶-۱

) کد مربوط به این بخش در فایل f.py قرار دارد. خروجی در تصویر ۷-۸ نمایش داده شده است.



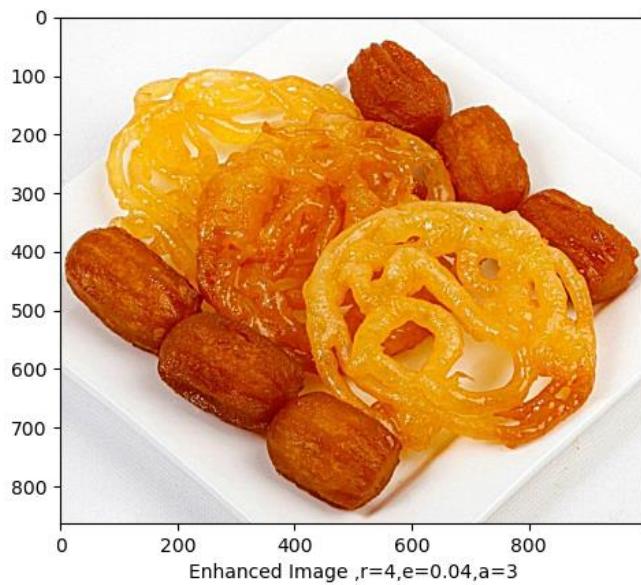
شکل ۱-۸

(g) از نظر کیفی تفاوت زیادی بین دو تصویر مشاهده نمی‌شود.

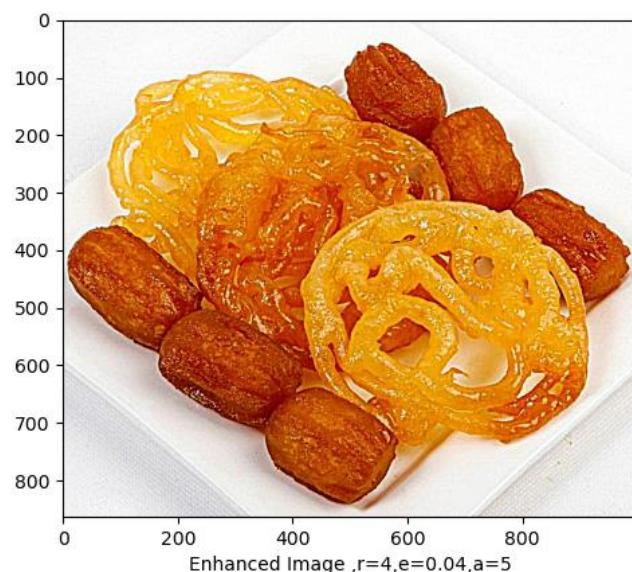
(h) کد مربوط به این بخش در فایل h.py قرار دارد. تصویر اصلی و تصاویر بهبود یافته به ترتیب در شکل‌های ۸-۸ تا ۱۱-۸ نشان داده شده است.



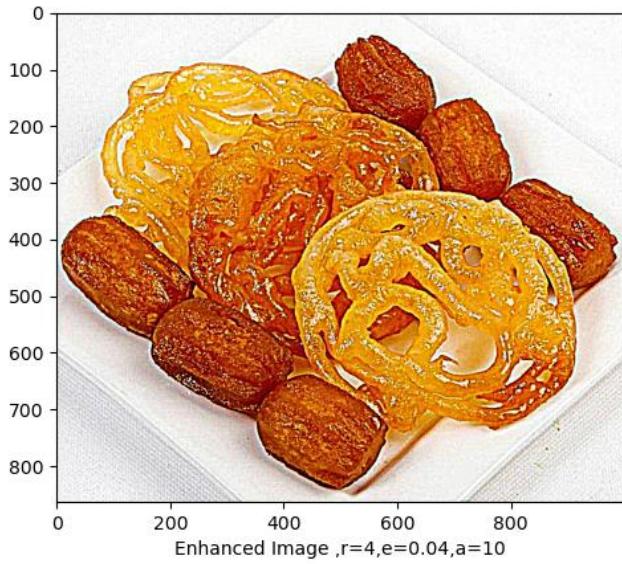
شکل ۱-۹



شكل ٤-١



شكل ٤-٢



(a) با توجه به معادله داده شده، افزایش a به وضوح باعث افزایش جزئیات می‌شود. از طرفی با توجه به قسمت a افزایش دو پارامتر دیگر باعث افزایش تصویر شده و در نتیجه جزئیات لبه‌ها کمتر می‌شود. بنابراین افزایش این دو پارامتر باعث کاهش جزئیات می‌شود.

(۹)

(a) بله، پس از اعمال متوالی این فیلتر، مرزها به بزرگترین مقدار و سایر پیکسل‌ها به کوچک‌ترین مقدار می‌رسند. در نهایت یک تصویر دودوبی خواهیم داشت که اعمال بیشتر این فیلتر تغییری در آن ایجاد نخواهد کرد.

(b) صفر کردن کم ارزش‌ترین بیت، موجب از بین رفتن مقادیر سطح خاکستری فرد در هیستوگرام خواهد شد. به تعداد پیکسل‌های فرد موجود در هر سطح به سطح زوج قبل از آن افزوده خواهد شد. این اتفاق تنها در صورتی که تعداد bin‌ها به تعداد سطوح خاکستری نزدیک باشد موجب تغییر در هیستوگرام خواهد شد. صفر کردن پرازش‌ترین بیت، موجب تغییر شدید و تیره شدن تصویر خواهد شد؛ در نتیجه در هیستوگرام نیز تغییرات شدید ایجاد خواهد شد و پیکسل‌ها در سمت چپ هیستوگرام جمع خواهند شد.

(c) 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.

(d)

(e) فرض کنیم هدف افزودن نویزی با میانگین صفر و واریانس ۱ به تصویر سطح خاکستری باشد. اگر به سه کanal تصویر رنگی، سه نویز با میانگین‌های صفر و واریانس‌های ۱ اعمال کنیم و سپس با استفاده از میانگین وزن دار سه کanal را با هم ترکیب کنیم داریم:

$$\begin{aligned} noisy_gray &= a \times (R + N) + b \times (G + N) + c \times (B + N) = (aR + bG + cB) + (a + b + c)N \\ &= gray + (a + b + c)N \end{aligned}$$

در رابطه فوق R ، G و B کanal‌های رنگی، N توزیع نرمال با میانگین صفر و واریانس ۱ می‌باشد. از طرفی می‌دانیم عبارت $(a + b + c)N$ یک توزیع نرمال با میانگین صفر و واریانس $\sigma^2 = a^2 + b^2 + c^2$ می‌باشد؛ بنابراین در این حالت نویز افزوده شده به تصویر دارای واریانسی متفاوت از ۱ می‌باشد و در نتیجه دو حالت گفته شده یکسان نیستند.