به نام خدا

گزارش پروژه اختیاری

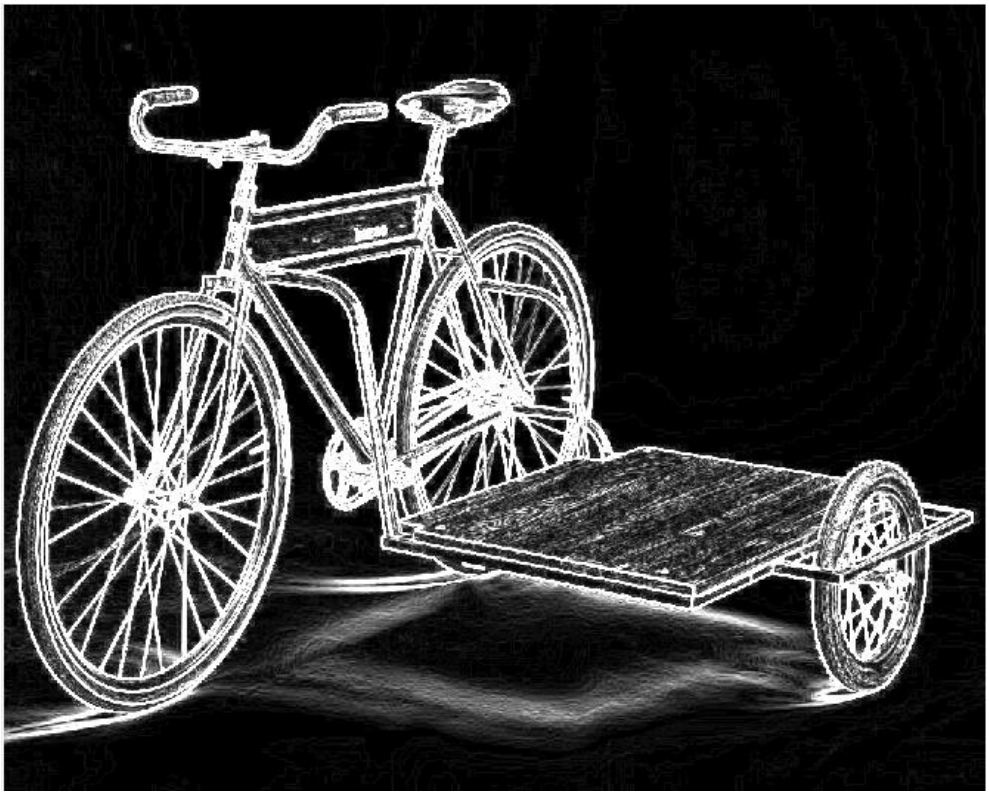
نگین اسماعیل زاده ۹۷۱۰۴۰۳۴

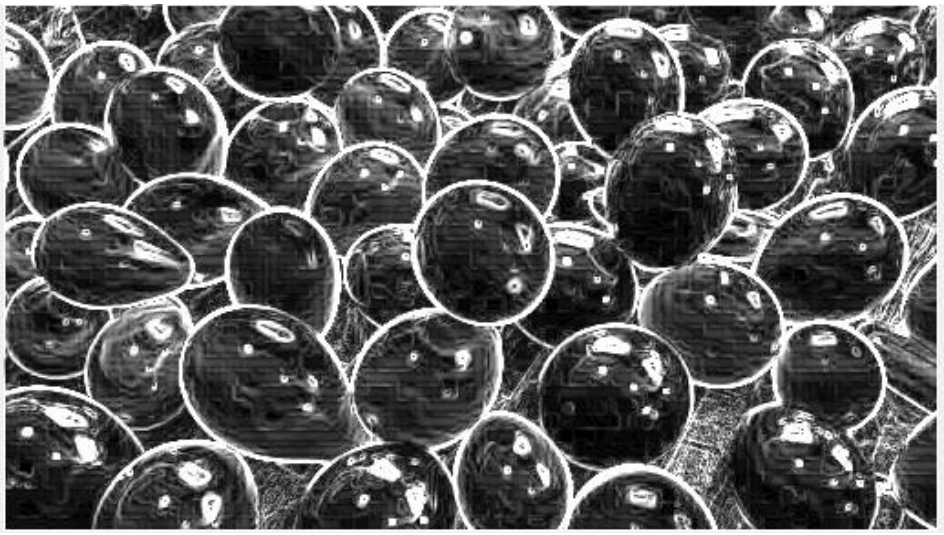
۱)

الگوریتم sobel :

زمان اجرای این الگوریتم برای مثال برای تصویر اول : T = 1.0562594

الگورتیم kirsch :





زمان اجرای این الگوریتم برای مثال برای تصویر اول : T = 1.4372974

مقایسه :

برای این بخش الگوریتم بهتر الگوریتمی است که بتواند edge ها را واضح تر و دقیق تر از بقیه قسمت های تصویر تفکیک کند ، با این معیار همانطور که میبینیم هر دو الگوریتم بسته به تصوری مورد برسی ما میتوانند مناسب باشند ، برای مثال الگوریتم kirsch چون مشتق گیری را در چند جهت انجام میدهد تصاویری با لبه های دایره و منحنی شکل را بهتر میتواند مشخص کند در حالی که الگوریتم sobel در مشخص کدن لبه های اشکلی ه تقریبا به صورت خط هستند(با مشتق گیری در دو جهت به خوبی شناسایی میشوند) تمیز تر عمل میکند .

به نظر میرسد زمان اجرای الگوریتم sobel از kirsch روی تصاویر یکسان کمتر است که با توجه به این موضوع که در واقع الگوریتم دوم ازلحاظ عمکرد بیشتر از الگوریتم اول عمل مشتق گیری را انجام میدهد انتظار این اتفاق را داشتیم .

۲) برای این کار روش زیر در نظر گرفته شده :

ابتدا تصویر را تصود یکی از الگوریتم های لبه یابی ، لبه یابی میکنیم ، سپس روی تمام پیکسل ها حرکت میکنیم و با فرض اینکه در هر پیکسل دایره عه به شعاع r وجود دارد (r از حد پایین تا حد بالا حرکت میکند) مجوع محتوای پیکسل هایی که تقریبا در فاصله شعاعی این پکسل قرار دارند را محاسبه میکند ، این مجموع مجموع میزان روشنایی است ، بعد از این مرحله یک حد برای تشخیص دایره قرار میدهیم که اگر مجموع روشنایی از آن حد بیشتر شد آن را به عنوان وجود مرکز یک دایره در این پیکسل تلقی کند سپسماتریس این مراکز را به تابع circle دادم ، وظیفه این تابع دسته کردن مراکز نزدیک هم است (چون در قمت قبل ممکن است نزدیک یک نقطه مرکزی تعداد زیادی دایره تشخیص داده شود).

نتیجه ی کد به وسیله دستورات متلب فقط در این قسمت مشاهده شد که برابر با ۲۶ دایره در تصویر بود

26

۳) انواع نویز :

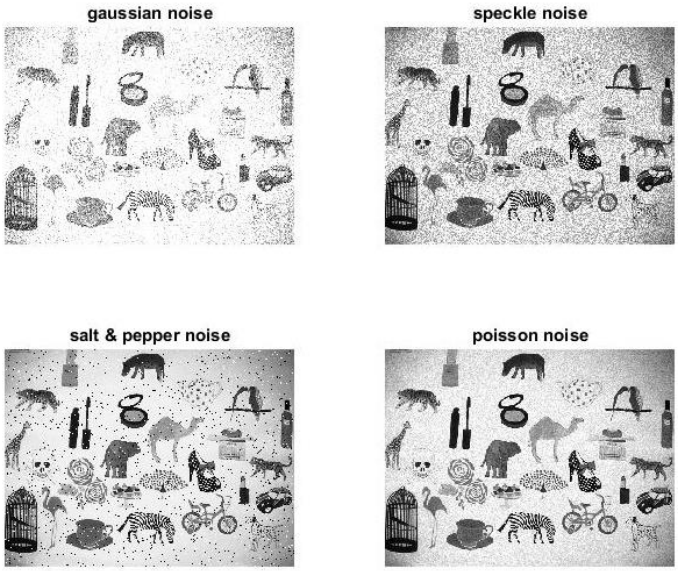
Gaussian (گاوسی) : این نویز از نوع نویز های جمع شونده (با سیگنال اصلی به صورت مقاویر مثبت با منفی جمع جبری میشود) است و تابع توزیع احتمال آن به صورت گاوسی است ، به این معنا که نویز های بیشتر با احتمال کمتری رخ می دهند اما نویز های بیشتر محتمل ترند . با توجه به توزیع ، فیلتر گاوسی برای حدف این نویز مناسب است .

Speckle (خال خالی) : این نویز ها ماهیت ضرب شونده دارند به این صورت که به جایجمع شدن با سیگنال اصلی در آن ضرب می شوند ، بنابریان اگر صفر باشند باعث میشود کل سیگنالمان صفر شود. (بیشتر در تصاویر راداری رخ میدهند و از مدل های رایج آماری نمیتوان برای حذ آنها استفاده کرد) . برای حذفاین نویز میتوان از فلتر میانگین یا میانه(تطبیقی یا غیر تطبیقی) استفاده کرد.

Salt and pepper (نمک و فلفل) : این نویز ها اکثرا در فرایند انتقال اطلاعات به وجود می آیند و به صورت فقط در دو مقدار احتمال رخ دادن دارند یا صفر یا ۲۵۵ (در تصویر ۸ بیتی (2^8-1)) و عملکرد انها به این صورت است که در صورت رخ دادن مقدار نویز را جایگزین مقدار سیگنال میکنند و بنابر این یا سیگنال را کاملا صفر میکنند یا ۱ و حالت میانی وجود ندارد . حذف این نویز ها به وسیله فیلتر میانه امکان پذیر است.

Poissen : همان طور که از اسم آن پیداست این نویز تابع توزیع احتمال پواسون دارد ، نویز پواسون یا نویز فوتون در اثر احتمالی بودن مسزا فوتون های ساطع شده از سمت سوژه به منبع تصویر برداری است که اندازه این نوع نویز وابسته به سیگنال است اما غیر از در شرایط کم نور عمده نویز تصویر را تشکیل میدهد.

منابع : Wikipedia , mapscale.ir , link.springer.com



فیتر گاوسی : فیلتر گاوسی فیلتری است که پاسخ ضربه ی آن نزدیک به تابع گاوسی باشد عملکرد آن به این صورت است که ماتریس تابع گاوسی را با ماتریس تصویر convolve میکند . برای مثال کرنل ۵x۵ تقریبی آن نیز به صورت زیر است :

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

همانطور که میبینم رفتار این ماتریس شبیه تابع گوسین هست به این صورت که اگر نقطه وسط ماتریس را به مکان (0,0) اختصاص دهیم هر چه از این مکان فاصله میگریم وزن تاثیر گذاری خانه های اطراف به تریجکمتر میشود ، پس اگر این فیلتر را روی تصاویر اعمال کنیم میانگین وزن دار خانه های اطراف را جایگزین هرخانه

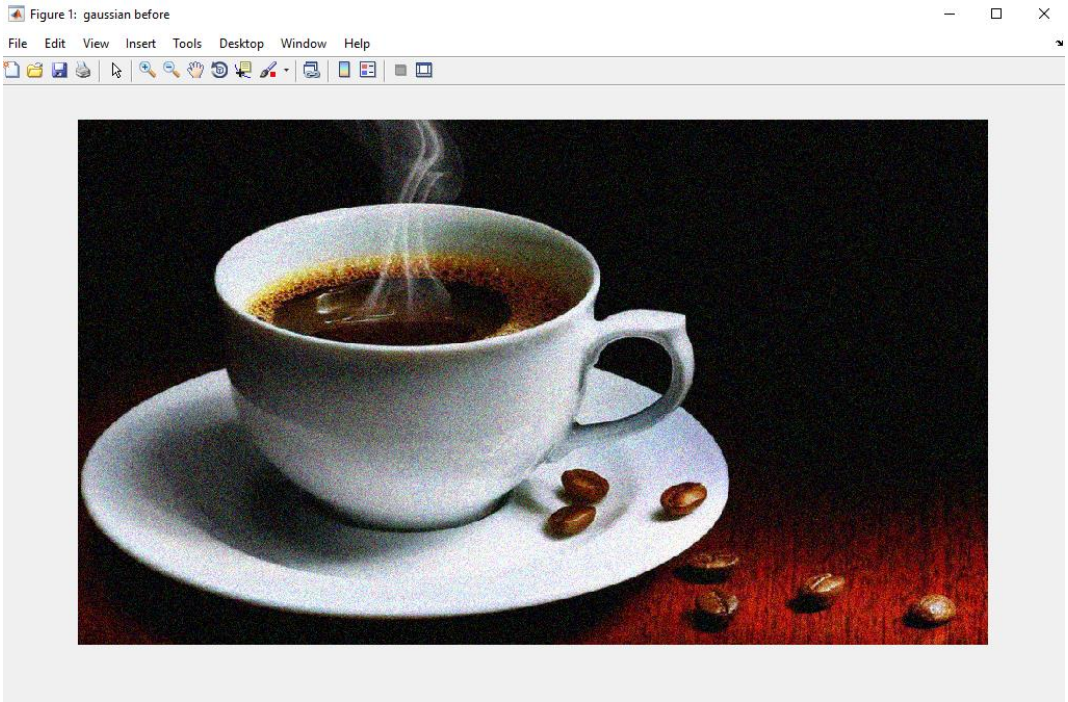
میکند،توزیع این وزن ها به طوریست که خانه میانی نقش اصلی را بازی میکند . البته برای محدود ماندن انرژی هنگام استفاده این ماتریس ها آن را به مجوع مقادیر خانه ها نیز تقسیم میکنم(انرژی تابع گوسی ۱ است).

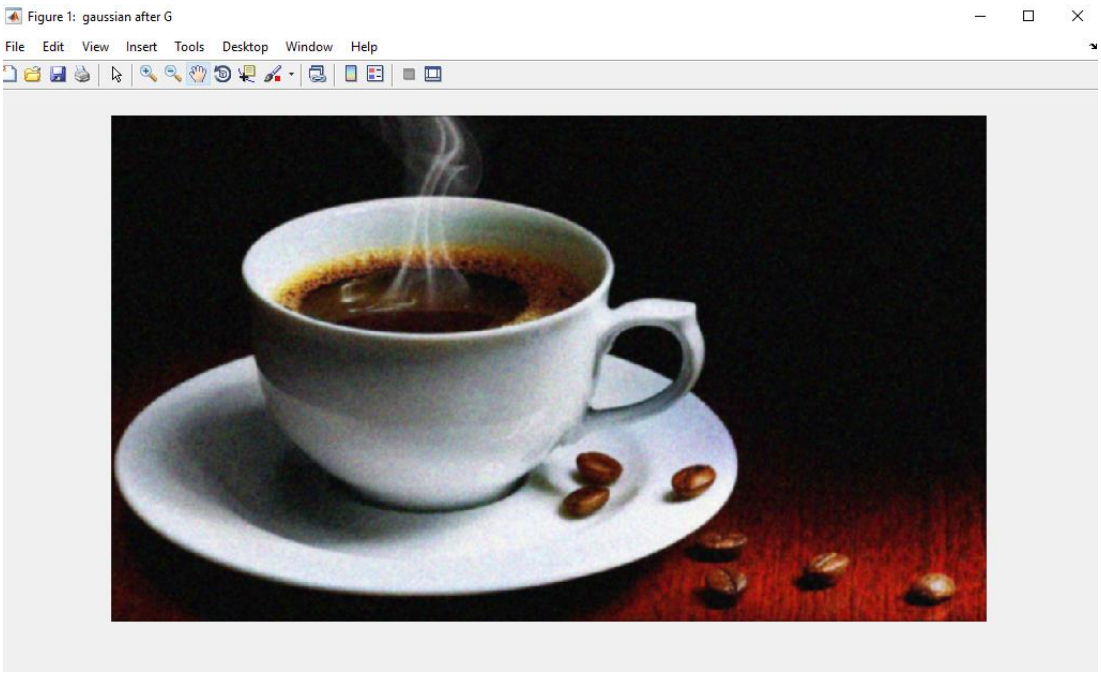
فیلتر میانه : الگوریتم کلی فیلتر میانه با پیدا کردن میانه و جایگزینی آن در مجموعه ورودی کار می‌کند.

کاربرد اصلی فیلتر میانه یک‌دست‌سازی تصویر و گرفتن نویزهای ضربه‌ای است. جایگزین کردن میانه در هر پنجره عکس باعث می‌شود تا مقادیری که از میانه دورترند حذف شوند و عکس به چیزی که بیشتر در آن تکرار شده شبیه می‌شود، می‌توان گفت که فیلتر میانه به بلوری شدن عکس کمک می‌کند و جزئیات را از بین می‌برد.

فیلتر میانگین : به این صورت است که وزن تمام درایه های اطراف یک خانه در جاگزین کردن امها خانه یکسان است ، در واقع هر خانه با میانگین اطرافیانش جایگزین میشود . در همین مرحله تفاوت دو فیلتر میانه و میانگین پیداست برای مثال در تصویریکه نویز نمک و فلفل دارد فیلتر میانه از میانگین بهتر عمل میکند چون فیلتر میانه به احتمال زیاد خانه های نویز را تمام جذب میکند ولی فیلتر میانگین کمی از اثر پذیری بیشتری از خانه ها نویز دارد. ماتریس این فیلتر در ابعاد مختلف از درایه های تماما یک تشکیل شده و بر ضریب مناسب مانند باقی فیلتر ها تقسیم میشود.

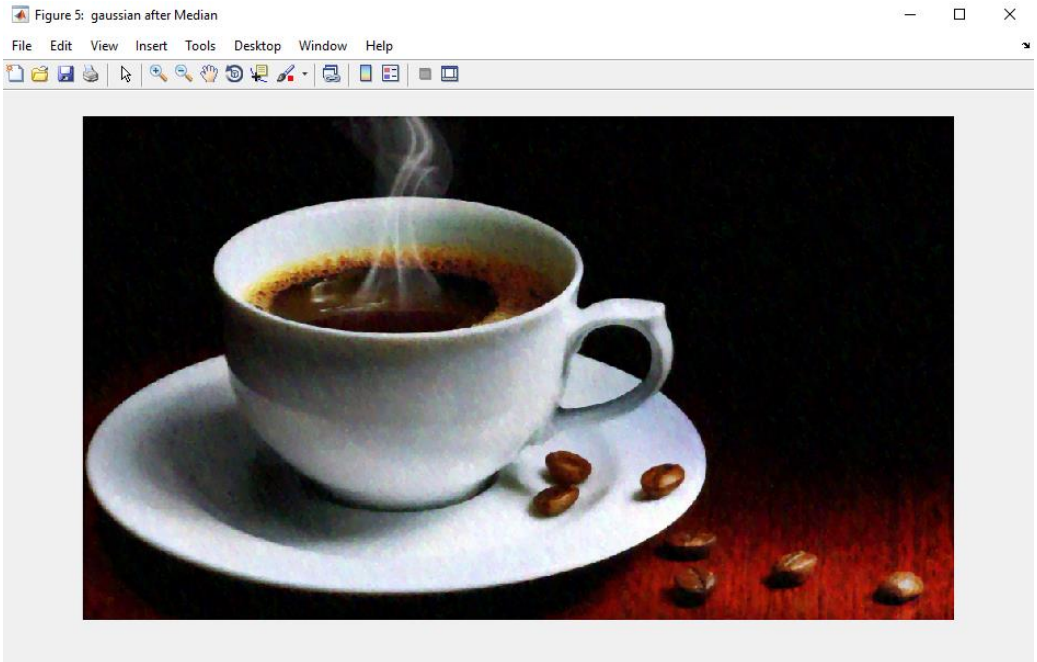
تصویر دارای نویز گاوسی و بعد از اعمال فیلتر ها





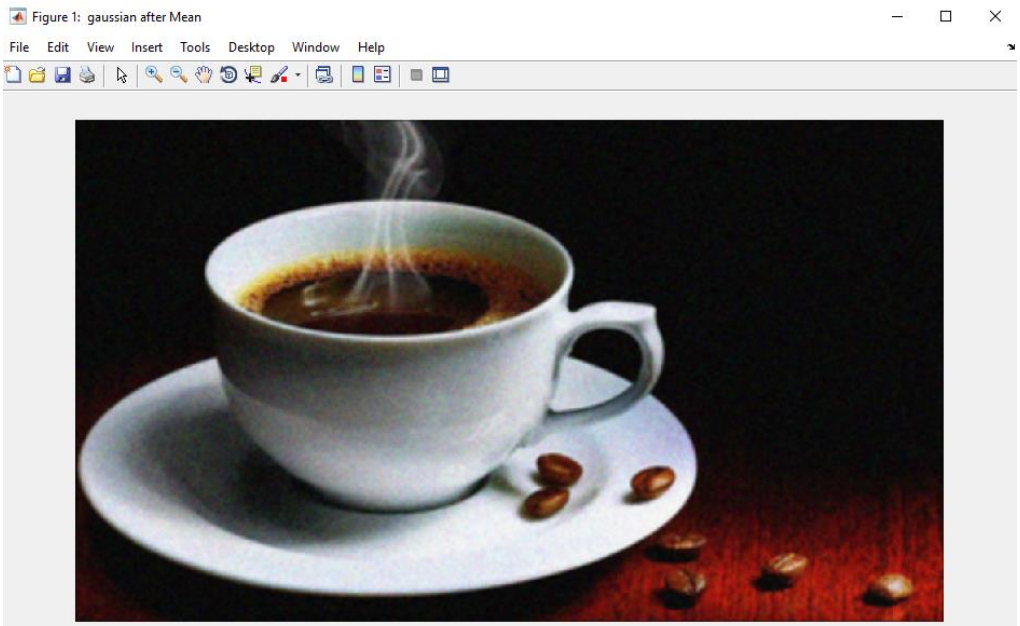
فیلتر گاوسی در این حالت بنظر بهترین حالت بود : n =11 ، sigma = 1.2

R = 0.9961



فیلتر میانه در این حالت بنظر بهترین حالت بود : n =3

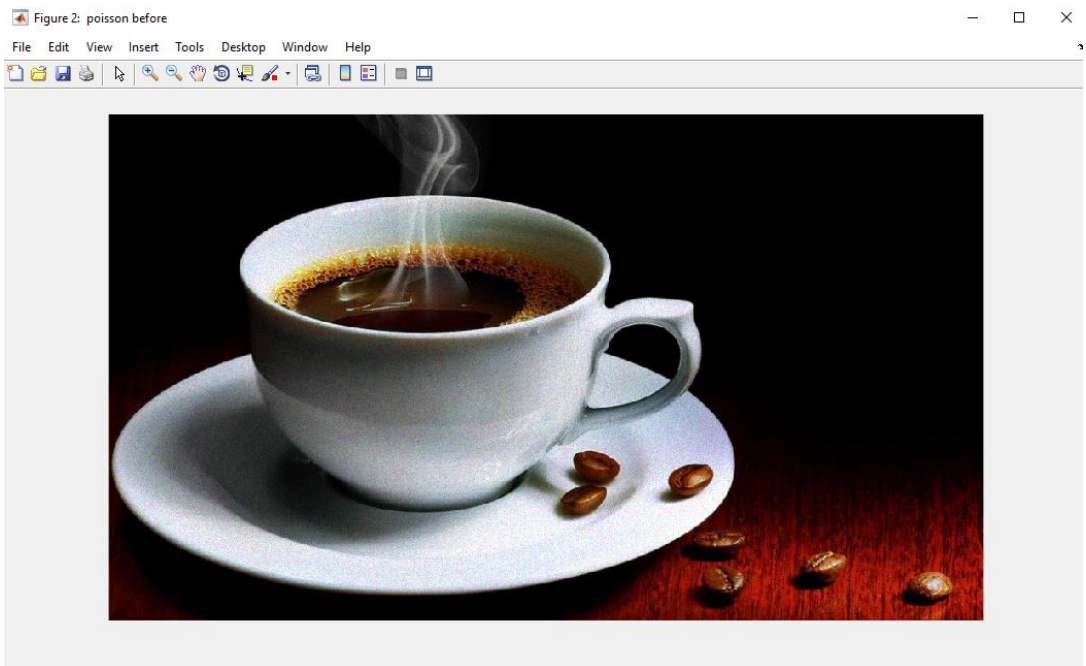
R = 0.9961

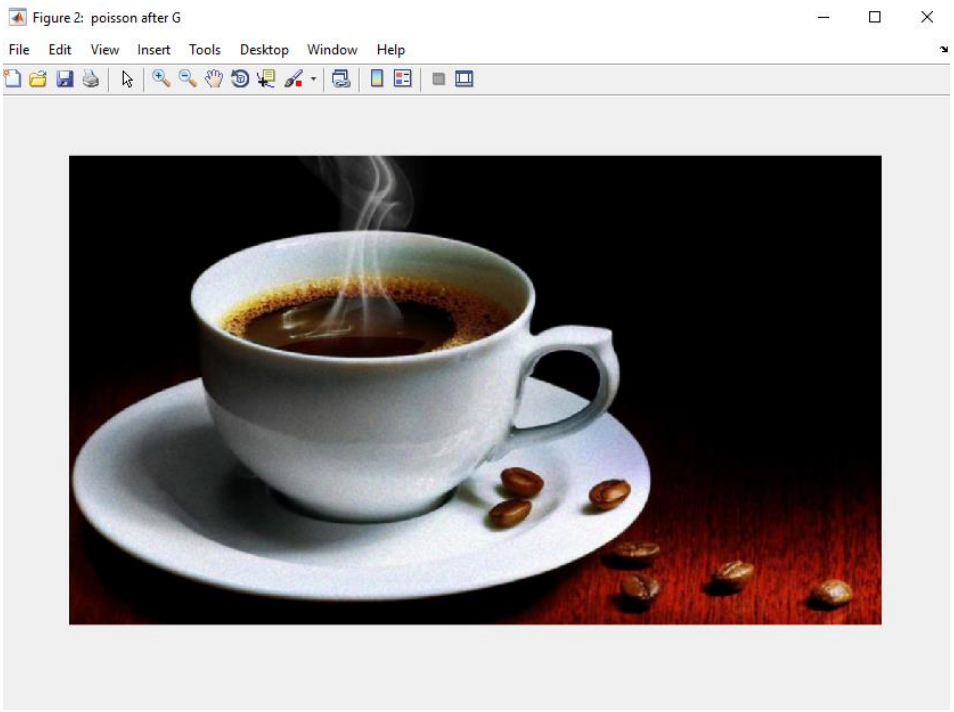


فیلتر میانگین در این حالت بنظر بهترین حالت بود : n =5

R = 0.9953

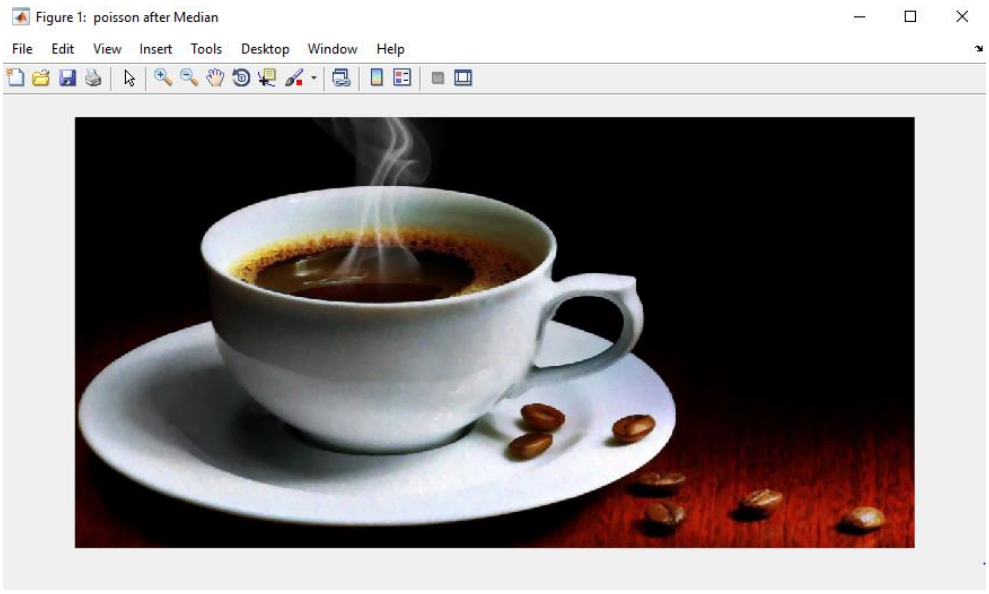
تصویر دارای نویز پواسون و بعد از اعمال فیلتر ها





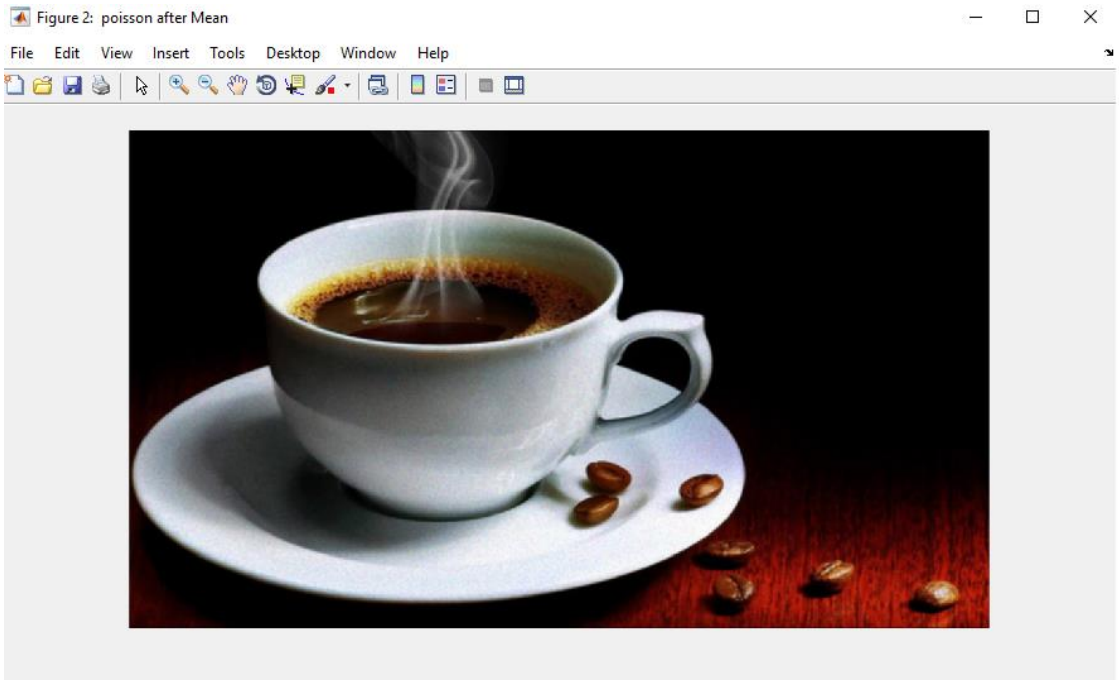
فیلتر گاوسی در این حالت بنظر بهترین حالت بود : n =3 ، sigma = 0.8

R = 0.9988



فیلتر میانه در این حالت بنظر بهترین حالت بود : n =3

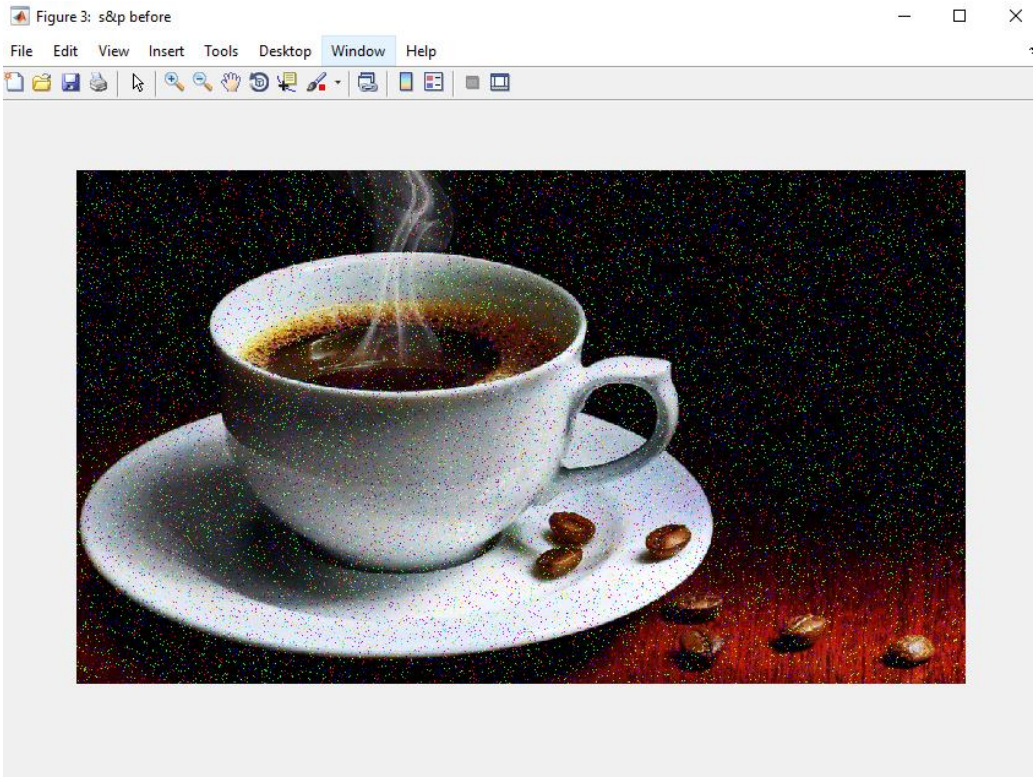
R = 0.9987

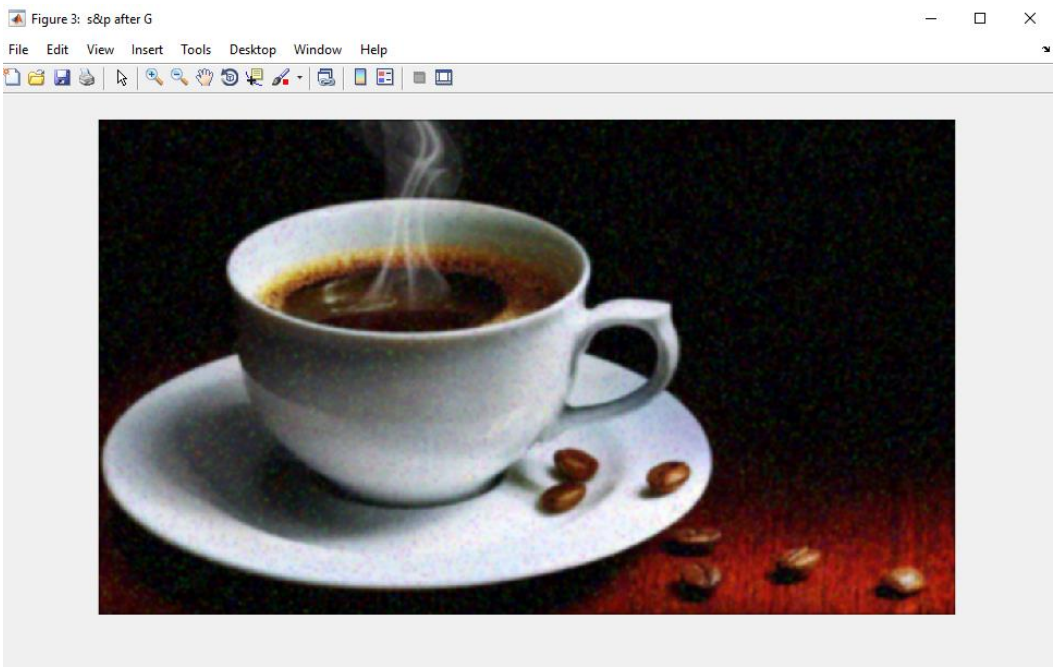


فیلتر میانگین در این حالت بنظر بهترین حالت بود : n =3

R = 0.9985

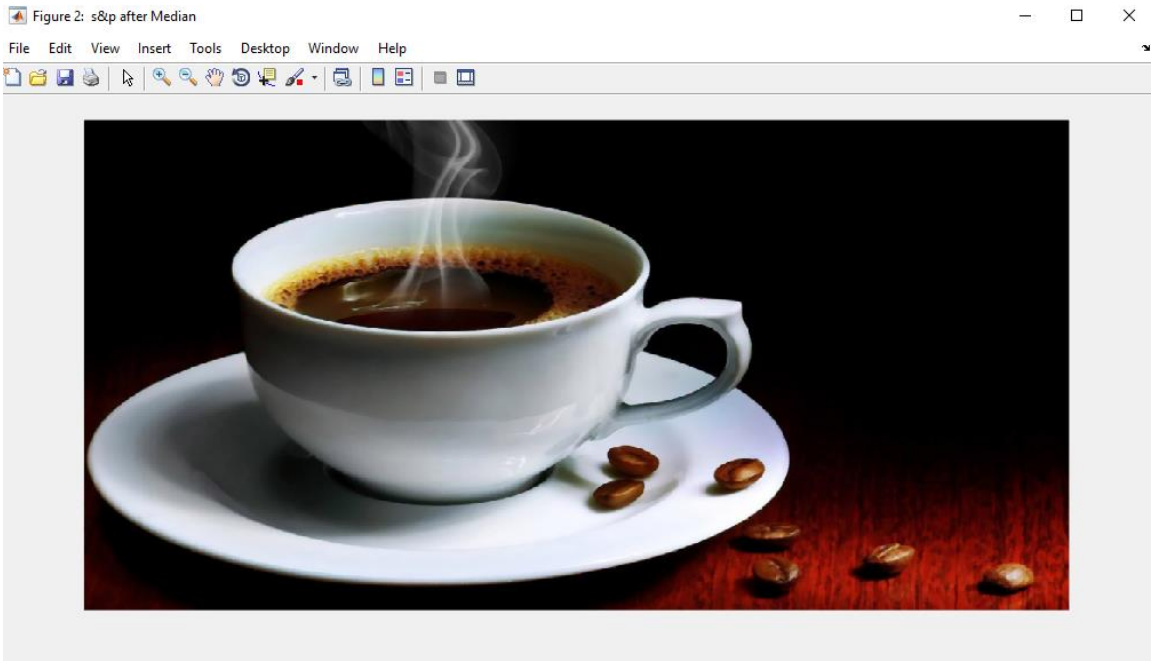
تصویر دارای نویز نمک فلفل و بعد از اعمال فیلتر ها





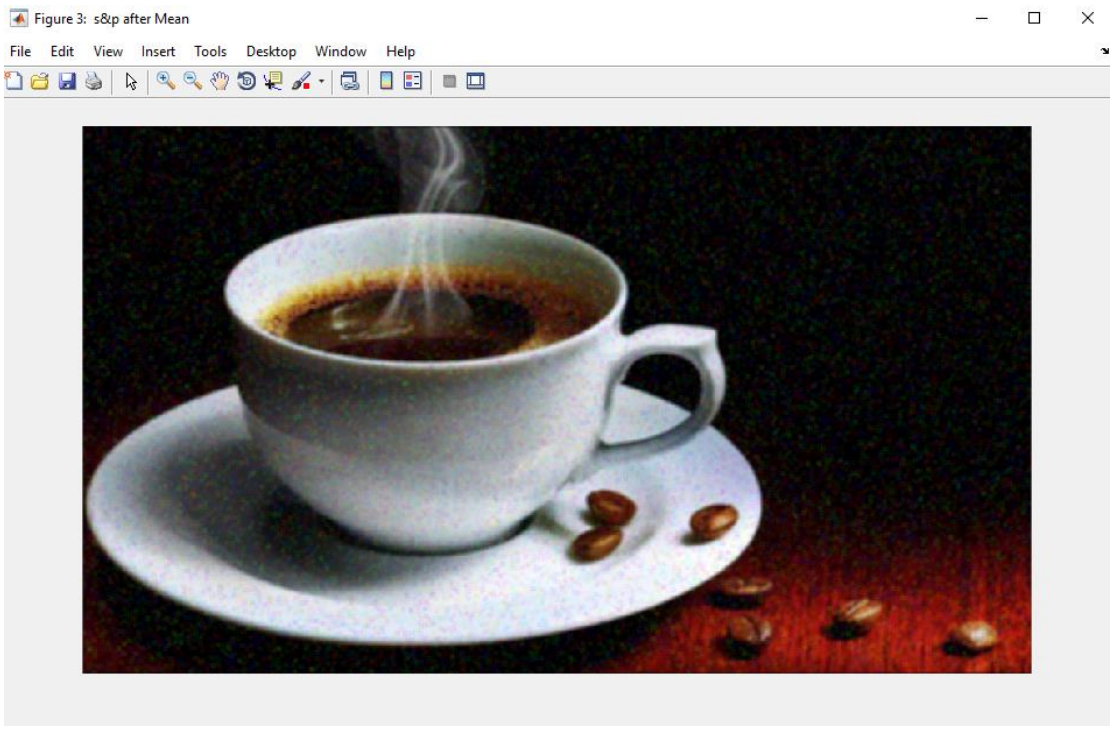
فیلتر گاوسی در این حالت بنظر بهترین حالت بود : n =7 ، sigma = 1.8

R = 0.9938



فیلتر میانه در این حالت بنظر بهترین حالت بود : n =3

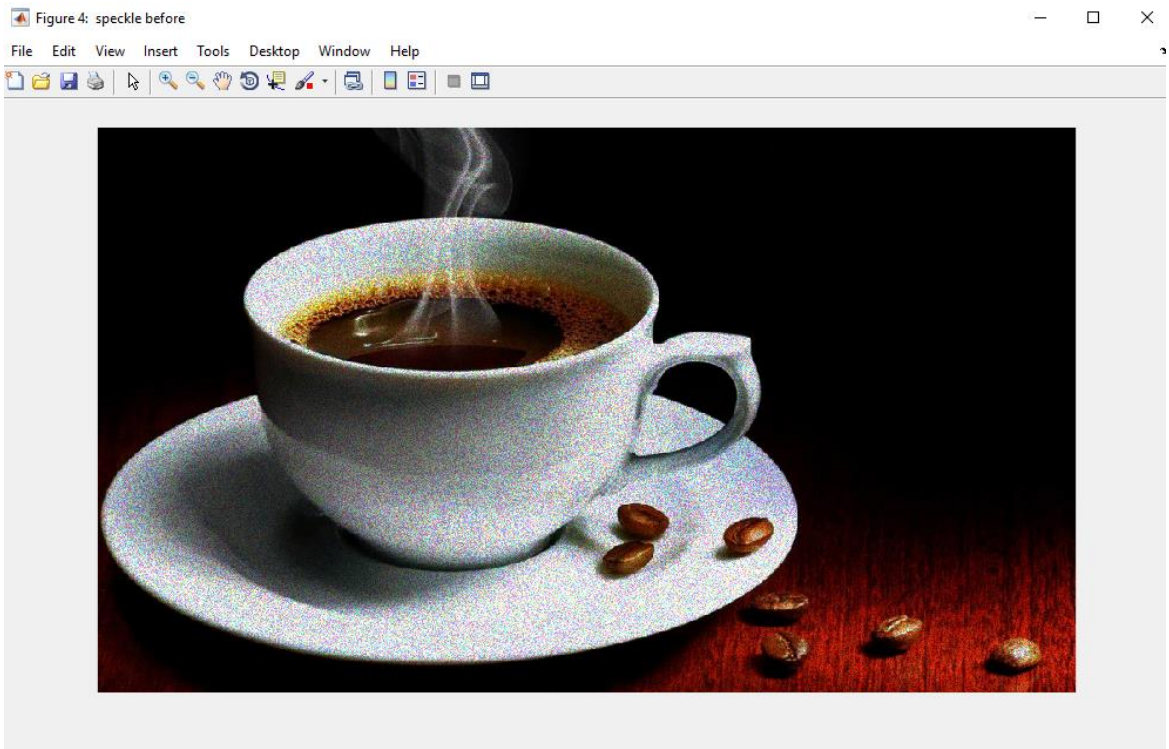
R = 0.9993

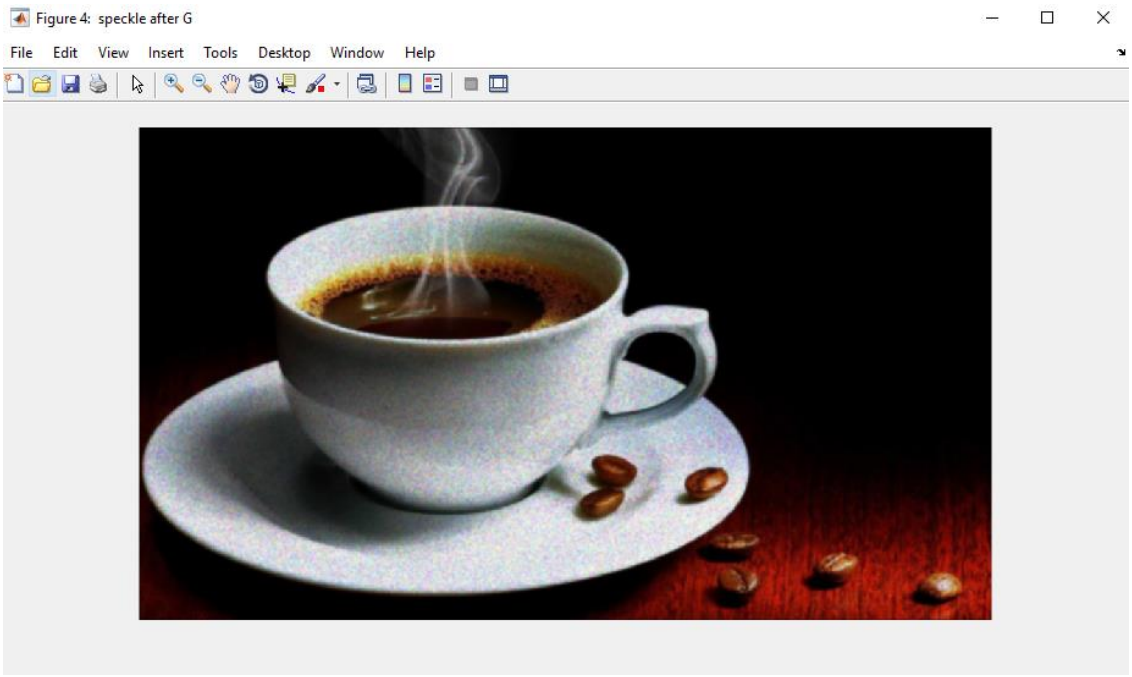


فیلتر میانگین در این حالت بنظر بهترین حالت بود : n =5

R = 0.9926

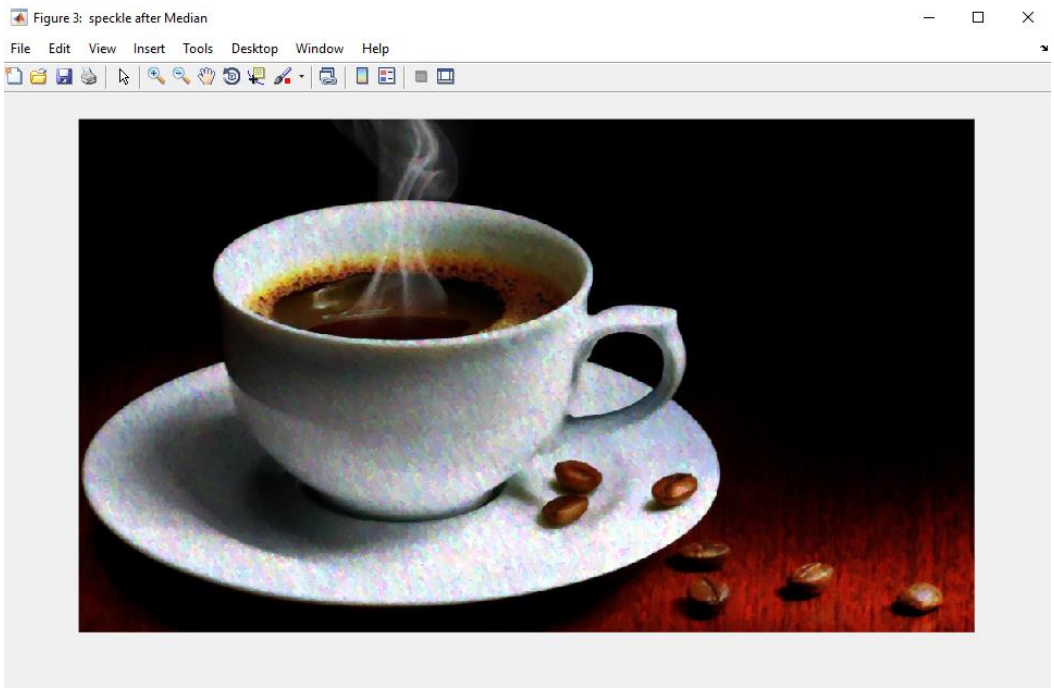
تصویر دارای نویز خالخالی و بعد از اعمال فیلتر ها





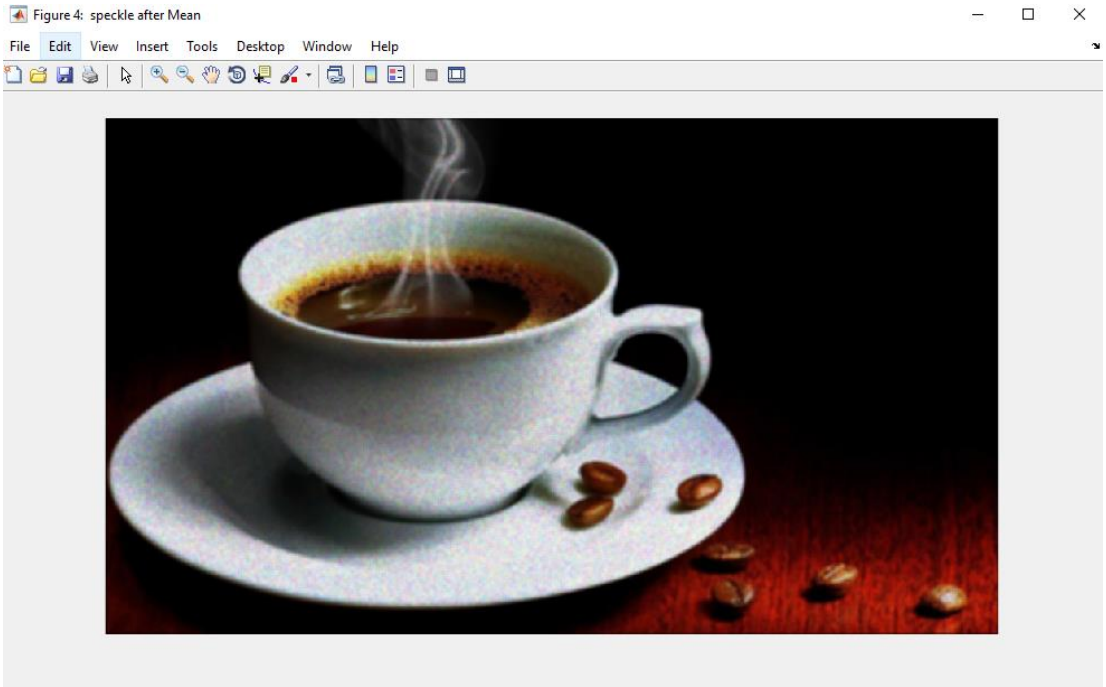
فیلتر گاوسی در این حالت بنظر بهترین حالت بود : n =9 ، sigma = 1.2

R = 0.9959



فیلتر میانه در این حالت بنظر بهترین حالت بود : n =3

R = 0.9942



فیلتر میانگین در این حالت بنظر بهترین حالت بود : n =5

R = 0.9950

نتایج استخراج شده از متلب بعد از هر فیلتر (چون احتمالا اجرا کردن کد کمی زمان ببرد)

فیلتر گاوسی

ng	11
np	3
ns	7
nsp	9
rg	0.9961
rp	0.9988
rs	0.9959
rsp	0.9938
s	10
sigmag	1.2000
sigmap	0.8000
sigmas	1.2000
sigmasp	1.8000

فیلتر میانه

ng	3
np	3
ns	3
nsp	3
rg	0.9961
rp	0.9987
rs	0.9942
rsp	0.9993

فیلتر میانگین

ng	5
np	3
ns	5
nsp	5
rg	0.9953
rp	0.9985
rs	0.9950
rsp	0.9926

در نتیجه این برسی متوجه میشویم برای حذف نویز گاوسی و پواسون فیلتر گوسی ومیانه فیلتر های بهتری هستند و در بیین این دو گاوسی بهتر است است ، برای حذف نویز نمک و فلفل فیلتر میانه بهترین فیلتر است برای حذف نویز های خالخالی نیز فیلتر های میانگین و گاوسی خوب هستند.

(۱.۴

۱) عبارت داده شده در واقع تشکیل شده است مجموع وزن دار (میزان فواصل با تان دو ظاهر میشوند بنابراین تاثیر فواصل دورتر بیشتر است) فواصل نقاط درون هر خوشه از میانگین آنهاست ، طبیعتا اگر نقاط به اندازه کافی به هم نزدیک (شبیه) باشند فاصله دورتیرن انها از میانگین نیز کم میشود و اگر ما بتوانیم این پارامتر را برای همه ی دسته ها مینیمم کنیم توانستیم خوشه هایی با اعضای داره بیشترین شباهت به همداشته باشیم .

۲) الگوریتم متدوالی که در این زمینه استفاده میشود الگوریتم لوید است که مراحل زیر را دارد :

k نقطه را به عنوان مراکز اولیه در نظر می گیرد (که معمولا این نقاط رندم انتخاب میشوند برای همین همگرایی درست در این روش قابل تضمین نیست)
هر مقداری را که مشاهده میکند به نزدیک ترین خوشه (نزدیک ترین مرکز) اختصاص میدهد

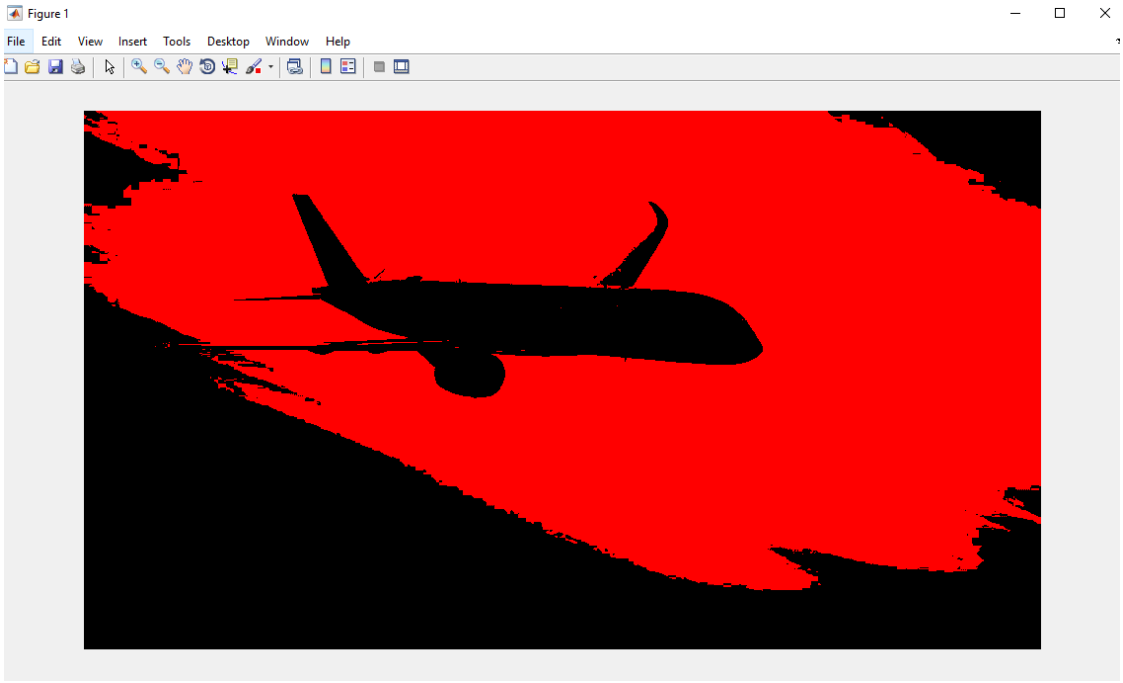
میانگین ها (مراکز) را مجددا محاسبه میکند و الگوریتم را آنقدر تکرار میکند تا زمانی که خوشه ها دیگر تغییر نکنند ، به عبارتی مراکز ثابت بمانند (شرط توقف)

۳) الگوریتم kmean الگورتیمی برای خوشه بندی به طریق یافتن نقاط با مینیمم فاصله از هم است، در این مسیله نیز ما میخواهیم پیکسل هایی از تصویر که به لحاظ رنگ و یا حتی فاصله مختصاتی به هم نزدیک هستند را در یک دسته قرار دهیم ، بنابراین برای این مسیله اگر الگوریتم را برای $k = 2$ انجام دهیم و با معیار وزن داری از فاصله و رنگ ها میتوانیم بخش بندی تصویر به ۲ بخش را انجام دهیم .

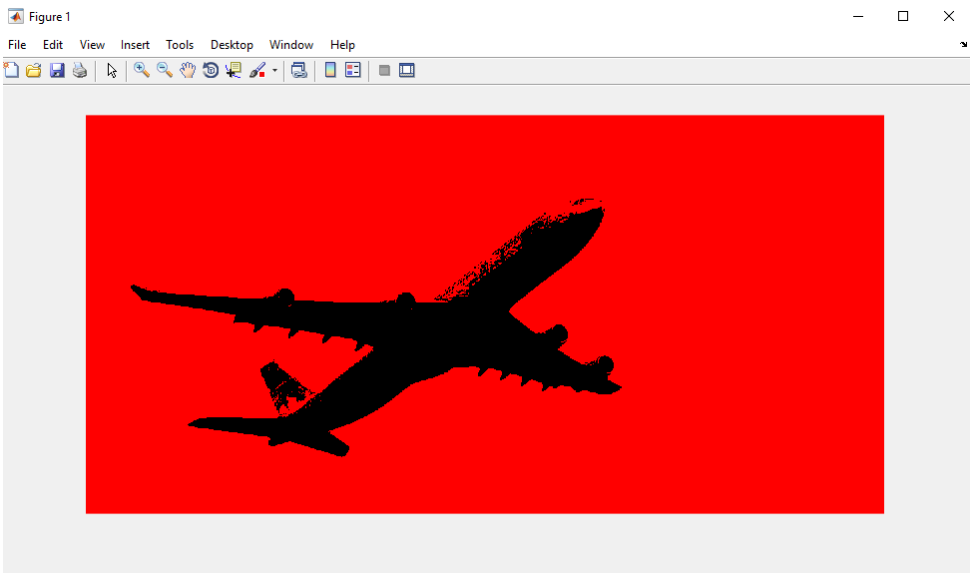
۴) تابع kmean که در کد موجود است مراحل فوق را انجام میدهد ، البته در این تابع فرض شده خوشه ها لزوما دو عدد هستند . در این تابع ابتدا تصویر را (چون محاسبات در حالت دو بعدی زمان خیلی زیادی میبرد) به دو بعد تبدیل کردم سپس عملیات تا همگرا شدن خوشه ها مطابق توضیحات بالا پیش برده شد.

متریک فاصله در این تایع بردار۳ بعدی ۳ رنگ است .

تصویر اول با ۱۵ بار انجام عملیت همگرا شد .(همانطور که در تصویر میبینیم این الگوریتم خیلی دقیق نیست به این دلیل که قسمتی از اسمان که هم رنگ هوایپیما بوده است در دسته هوا پیما قرارگرفته است ، البته برای بهتر شدن این موضوع میتوان اثر فاصله پیکسلی را نیز وارد کار کرد.)



تصویر دوم با ۱۸ بار انجام عملیات هگمرا شد.



۵)روش Otsu : اساس این عملیات بر پایه مشخص کردن یک حد معین برای روشنایی که از این طریق شکل را به دو قسمت کاملا روشن یا کاملا تاریک تقسیم کند است و این کر را از طریق مینیم کردن میانگین وزن دار واریانس های خوشه ها انجام میدهد که این وزن ها هم احتمال رخ دادن هر کدام هستند . این رابطه را با عملیات ریاضی به ماکزیمم کردن عبارت زیر تبدیل میکنند.

$$\omega_0(t)\omega_1(t)[\mu_0(t)-\mu_1(t)]^2$$

$$\omega_0(t)=\sum_{i=0}^{t-1}p(i)$$

$$\omega_1(t)=\sum_{i=t}^{L-1}p(i)$$

$$\mu_0(t)=\frac{\sum_{i=0}^{t-1}ip(i)}{\omega_0(t)}$$

$$\mu_1(t)=\frac{\sum_{i=t}^{L-1}ip(i)}{\omega_1(t)}$$

$$\mu_T=\sum_{i=0}^{L-1}ip(i)$$

$$\omega_0\mu_0+\omega_1\mu_1=\mu_T$$

$$\omega_0+\omega_1=1$$

و در این مرحله الگوریتم مراحل زیر را برای ماکزیمم کردن عبارت فوق طی میکنند.

ابتدا توزیع فراوانی و احتمال مربوط به هر شدت را محاسبه میکنیم.

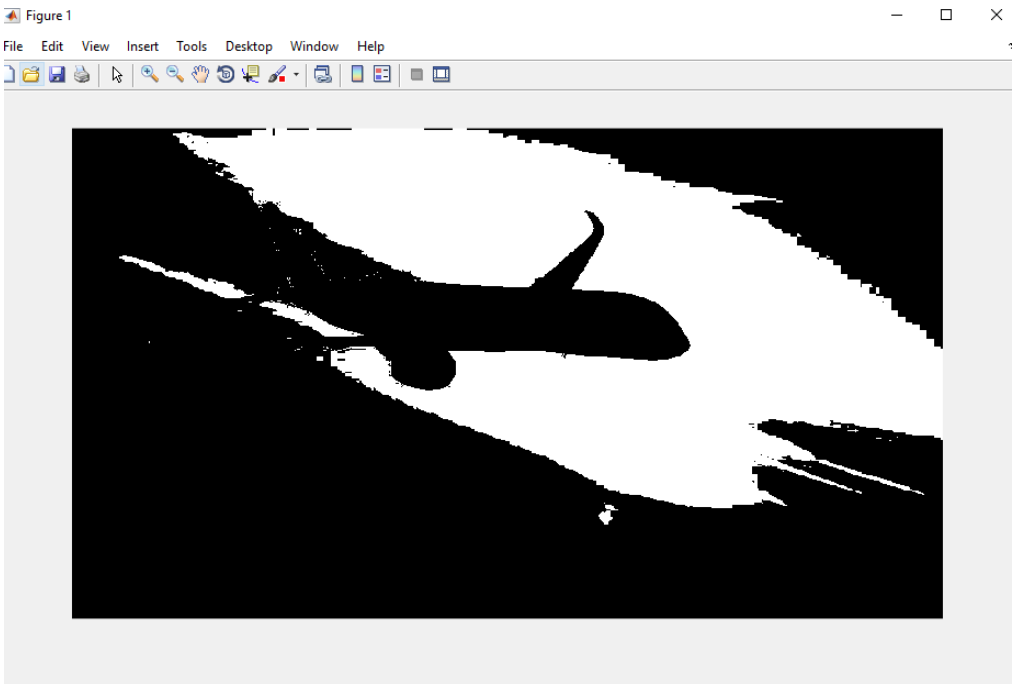
سپس M و W اولیه ای تشکیل میدهیم.

به ازای ترشهولد های مختلف روشنایی(یک حد معین از روشنایی که به توان به وسیله ان شکل را بخشبدی کرد) رابطه های ذکر شده را مجددا محاسبه میکنیم و مقدار عیبارتی که میخواهیم ماکزمم شود را بدست می اوریم .

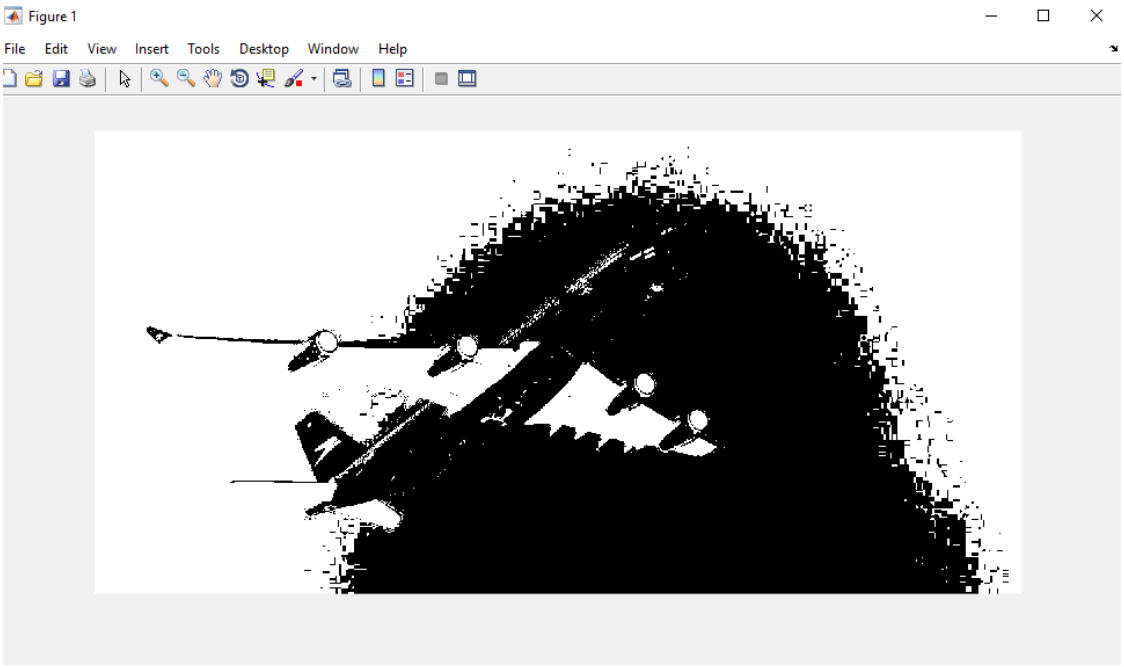
در نهایت حدی از روشنایی که باعث ماکزیمم شدن عبارت (به نوعی اختلاف میزان روشنایی دو دسته) شد را استخراج میکنیم و به هر پیکسل با توجه به این که مقدار روشنایی آن از این حد بیشتر یا کمتر است مقدار مربوط به کاملا روشن یا کاملا تیره اختصاص میدهیم.

برای این بخش در کد از تابع imhis استفاده شده است که هیستوگرام یا فراوانی هر مقدار روشنایی در تصویر را در خروجی میدهد.

تصویر اول با ترشهولد ۱۶۳ :



تصویر دوم با ترشهولد ۱۴۰

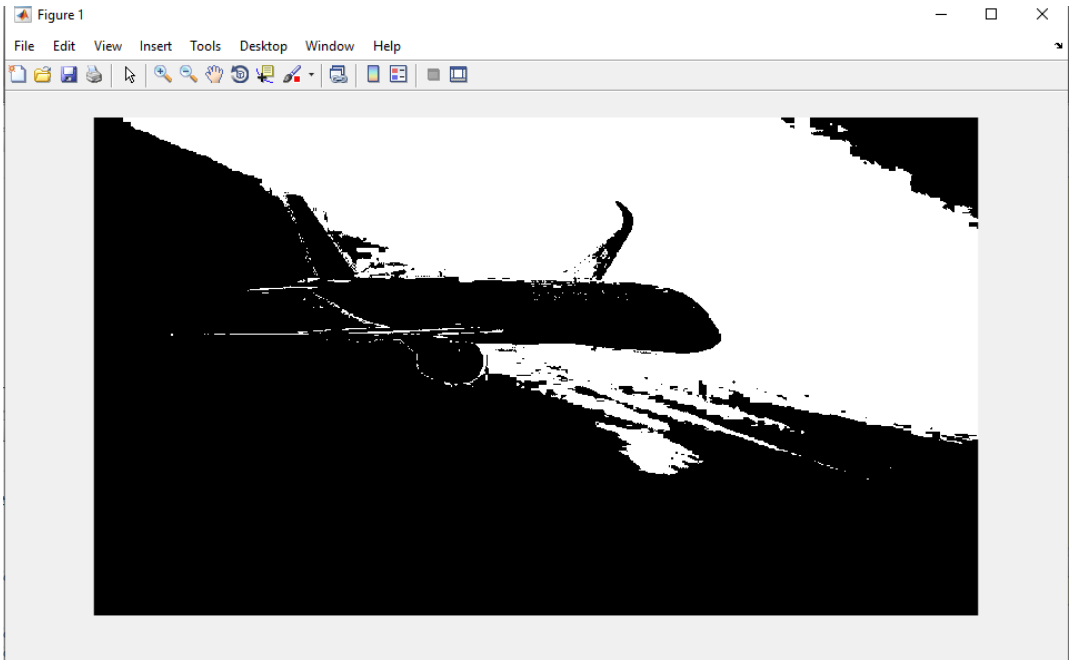


مقایسه :

از مقایسه خروجی ها متوجه میشویم روش kmean احتمالا به دلیل در نظر گرفتن دقیق تر فاصله رنگ ها خرجی واضحتری دارد ، اما از نظر زمان محاسبه روش Otsu به مراتب سریع تر است چون شامل یک حلقه به تعداد ۲۵۶ بار است و با افزایش سایز تصویر این زمان تقریبا تغییر نمیکند ، در حالی که زمان و تعداد حلقه های روش k mean وابسته به سایز تصویر است (و اگر بردار متریک را ترکیبی از فاصله و رنگ در نظر بگیریم این وابستگی بیشتر هم میشود) ، همچنین در حالی که روش kmean با توجه به میزان تصادفی بودن ممکن است همگرایی درستی نداشته باشد روش otsu همواره همگراست اگر چه احتمالا در صورت خروجی گرفتن از دو روش خروجی روش اول واضح تر است. منظور از دقت بیشتر این است که همانطور که دیددیم در پیاده سازی روش k mean عدم دقت خروی زمانی اتفاق میافتاد که رنگ قسمتی از بخش دوم شبیه بخش اول میشد و این باعث تشخیص اشتباه بود اما این رنگ شامل سه رنگ آبی قرمز و سبز بود ولی در روش دوم تنها روشنایی ملاک است بنابر این دقت این تفکیک کاهش پیدا میکند .

البته در تصاویر بالا روش Otsu را به تصویر خاکستری شده اعمال کردم ، اما اگر این موضوع را روی تک تک رنگ ها اعمال کینیم همانطور که در زیر مشاهده میکنیم نتیجه بهتر میشود (اگر چه همچان بسیار نادقیق است)

تصویر اول در این حالت :



تصویر دوم در این حالت: