به نام خدا

مبانی بینایی کامپیوتر

دکتر محمدی

تمرین دو

علی عطاریان – ۹۹۵۲۱۴۵۱

سوال ۱) الف)

کانوولوشن را مطابق این فرمول پیاده سازی می‌کنیم. در واقع در هر مرحله یک پنجره از عکس به اندازه ابعاد فیلتر جدا کرده و این پنجره را متناظرا در فیلتر ضرب کرده و جمع مقادیر این پنجره را در خروجی قرار می‌دهیم. در پیاده‌سازی بایستی به پدینگ و همچنین ایندکس مربوط به لوپ‌ها دقت کرد.

$$(w \star f)(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x-s, y-t)$$

در این سوال از فیلتر سوبل استفاده می‌کنیم که مقادیر آن مشخص است. همچنین فرمول اندازه و جهت گرادیان که از سوبل افقی و عمودی به دست می‌آید را نیز پیاده‌سازی می‌کنیم.

G_y

-1	-2	-1
0	0	0
+1	+2	+1

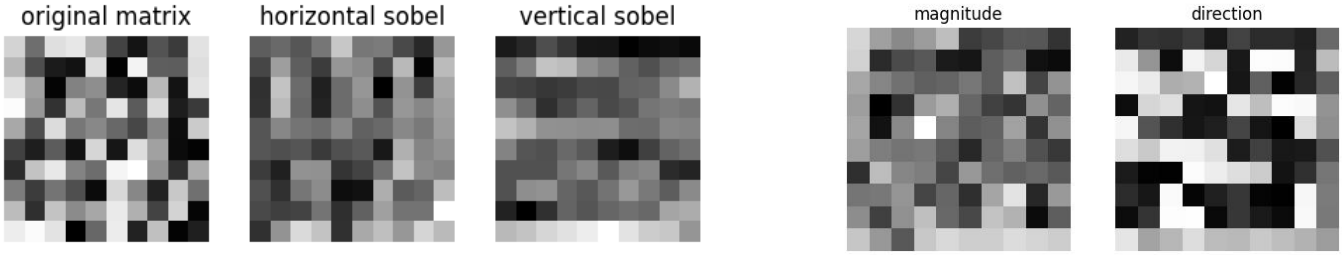
G_x

-1	0	+1
-2	0	+2
-1	0	+1

$$\text{mag} = \sqrt{g_x^2 + g_y^2}$$

$$\text{dir} = \text{atan2}(g_y, g_x)$$

خروجی بدین صورت است:

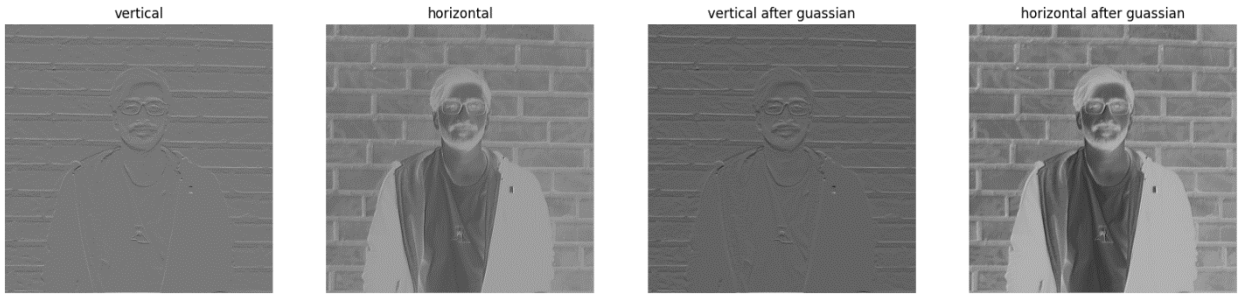


سوال ۱) ب)

فیلتر گاوس را مطابق فرمول ریاضی پیاده‌سازی می‌کنیم:

$$g(x,y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/(2\sigma^2)}$$

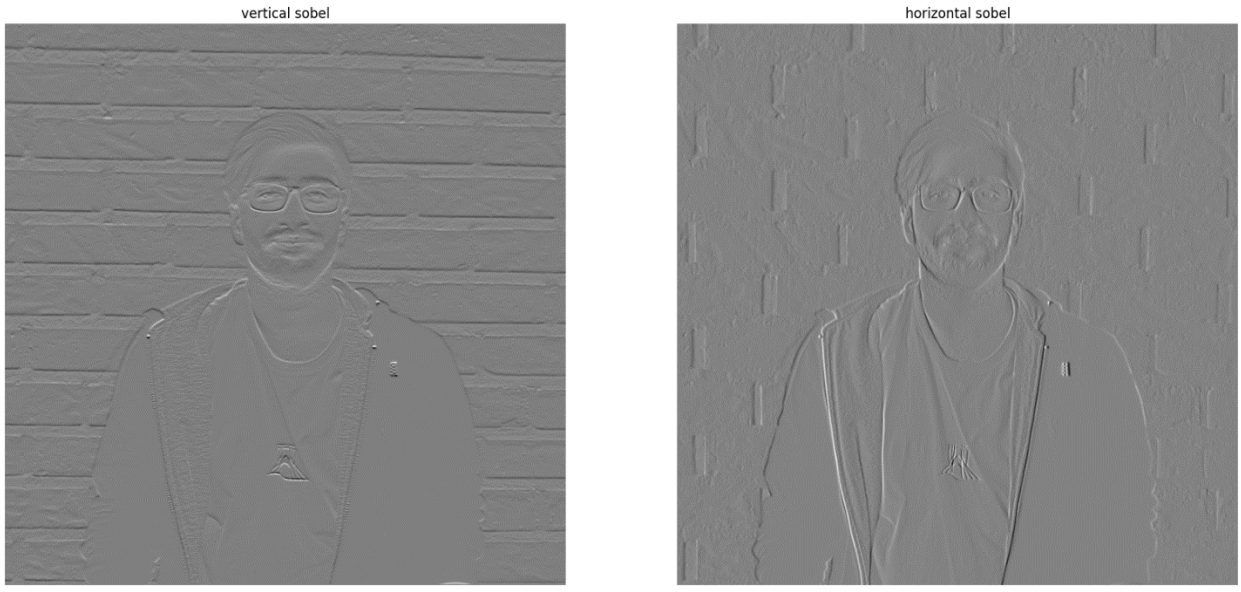
حال تصویر را با عملگر سوبل افقی و عمودی با/بدون فیلتر گاوسی پردازش می‌کنیم.



مشاهده می‌کنیم در تصاویری که سوبل بعد از فیلتر گوسی اعمال شده است، لبه ها واضح‌تر شناسایی شده‌اند.

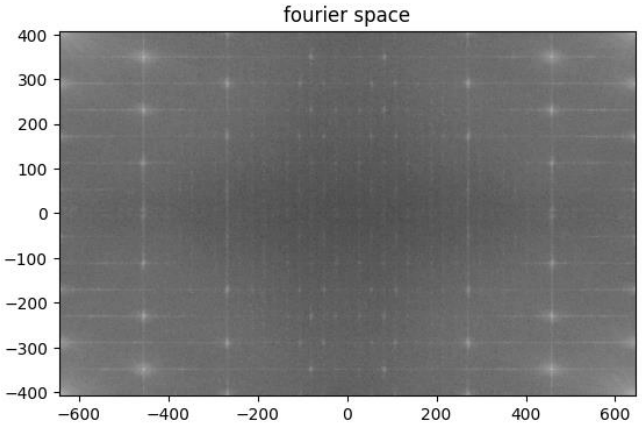
سوال ۱) ج)

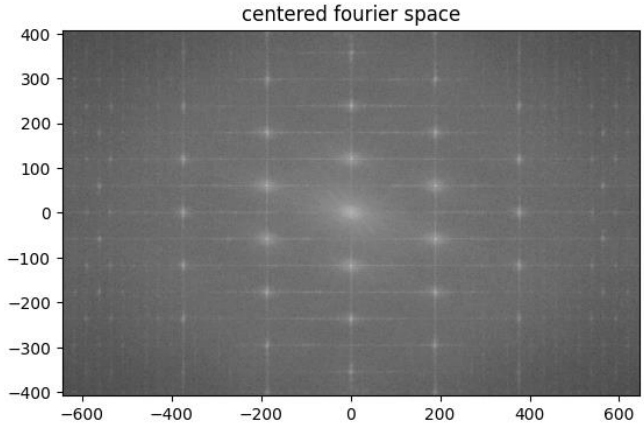
پارامتر های متد cv2.sobel به ترتیب تصویر ورودی، عمق تصویر خروجی که همان دیتاتایپ و تعداد کانال‌ها می‌باشد (در اینجا float و یک کاناله)، ست کردن مشتق افقی، ست کردن مشتق عمودی و سایز فیلتر می‌باشند. خروجی بدین صورت است:



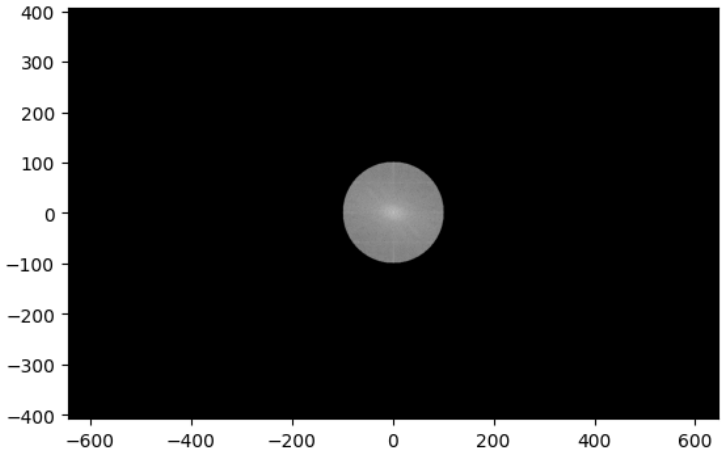
سوال ۲) الف)

ابتدا تصویر را با استفاده از np.fft.fft2 به فضای فوریه می‌بریم. سپس به اندازه نصف طول شیفت افقی و به اندازه نصف عرض شیفت عمودی می‌دهیم که با دستور np.fft.fftshift قابل انجام است. باید دقت داشت برای نمایش بصری فضای فوریه لگاریتم می‌گیریم زیرا در غیر اینصورت به دلیل مقدار بسیار زیاد در نقطه ۰و۰ تمام تصویر سیاه دیده می‌شود.

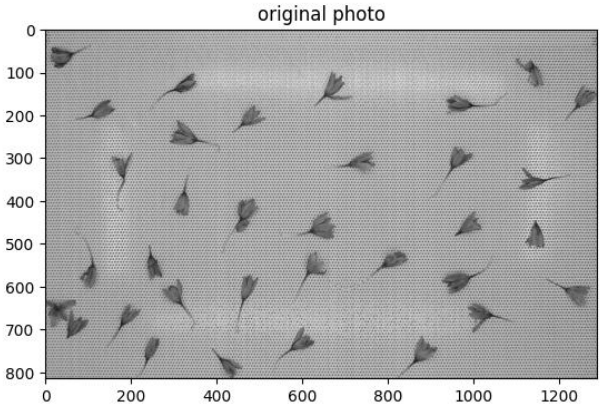


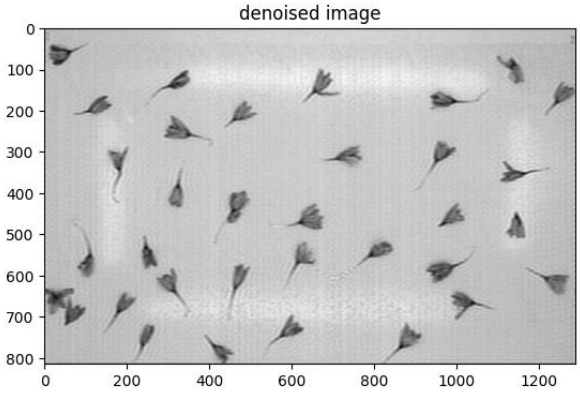


در تصویر centered fourier space پیک‌های روشنایی به جز پیک وسط تصویر نشان دهنده نویز می‌باشند. راه‌حل پیشنهادی برای از بین بردن نویز، نگه داشتن دایره‌ای کوچک در وسط و حذف باقی مقادیر است. با تابع denoisingFourier همه نقاط به جز دایره‌ای به مرکز تصویر و شعاع radius را سیاه می‌کنیم.



حال این ماتریس در فضای فوریه را با np.fft.ifftshift به حالت قبل از شیفت برگردانده و سپس با دستور np.fft.ifft2 به فضای تصویر می‌بریم تا تصویر دینویز شده را نمایش دهیم.

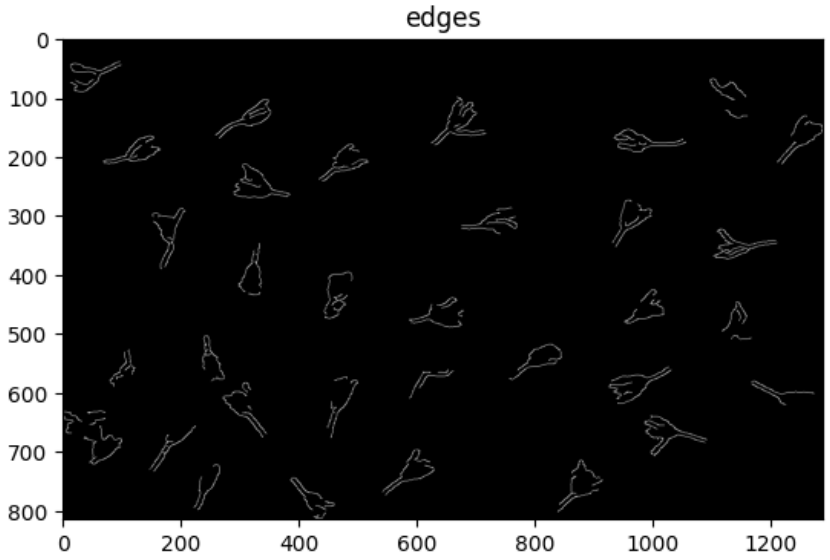




می‌بینیم روزنه‌ها که نویز بودند حذف شدند.

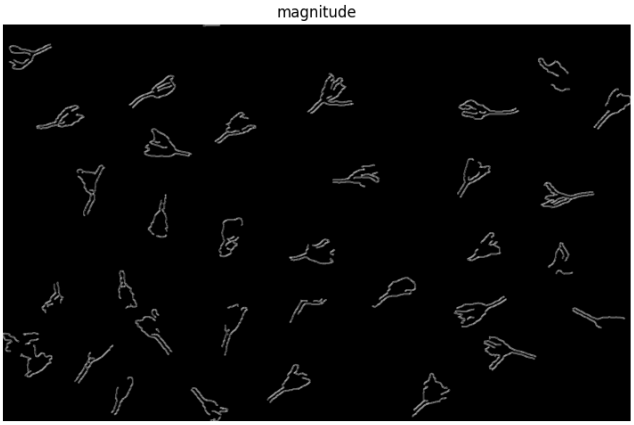
سوال ۲) ب)

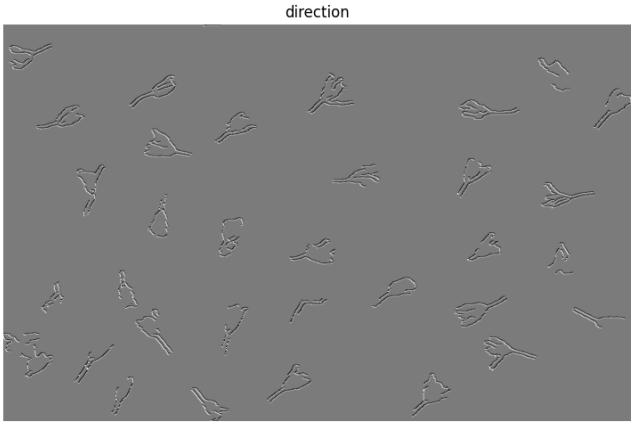
پارامترهای تابع cv2.canny به ترتیب تصویر ورودی، minValue و maxValue هستند. تصویر ورودی باید به تایپ uint8 کست شود تا ارور نگیریم. مقادیر بالاتر از maxValue حتما لبه هستند و مقادیر کمتر از minValue حتما لبه نیستند اما مقادیر بین این دو اگر به نقاط لبه(نقاط با مقدار بیش از maxValue) وصل باشند آنگاه لبه حساب می‌شوند. پس از کمی آزمون و خطا مقادیر مناسب برای این پارامترها ۵۰ و ۱۲۰ یافت شدند.



سوال ۲) ج)

اندازه و جهت گرادیان را مانند سوال یک محاسبه می‌کنیم.





سوال ۳) الف)

فیلتر پایین‌گذر فرکانس‌های پایین را عبور می‌دهد و تاثیر فرکانس‌های بالا کاهش می‌دهد یا حذف می‌کند به همین دلیل در تار کردن و smooth کردن تصویر و همچنین روش‌های کاهش نویز کاربرد دارد. فیلترهای میانگین و میانه و گاوسی مثالهایی از فیلتر پایین‌گذر هستند.

فیلتر بالاگذر فرکانس های بالا را عبور می‌دهد و تاثیر فرکانس های پایین را کاهش می‌دهد یا حذف می‌کند. از این فیلتر برای برجسته کردن لبه‌ها یا افزایش sharpness تصویر استفاده می‌شود. از فیلترهای بالاگذر می‌توان به فیلتر لاپلاسی و سوبل و پریویت اشاره کرد.

سوال ۳) ب)

از آنجایی که در این تصویر لبه ها برجسته شده‌اند پس از یک فیلتر بالاگذر استفاده شده است.

سوال ۳) ج)

نویز جمع‌شونده یعنی یک مقدار رندوم به مقادیر تصویر اضافه می‌شود که معمولا توزیع یونیفرم دارد و بر تمام سیگنال تاثیر می‌گذارد. نویزهای سفید، گاوسی و نمک‌فلفل از این نوع هستند. برای کاهش این نویز فیلتر میانه و فیلتر گاوسی پیشنهاد می‌شود.

نویز ضرب‌شونده نوعی نویز است که در تصویر اصلی ضرب می‌شود. علت آن می‌تواند تغییرات نوری یا مشخصات سنسور باشد. این نویز می‌تواند بخش‌های مختلف تصویر را به طرز متفاوتی تحت‌تاثیر قرار دهد و یونیفرم نیست. نویز نقطه‌ای (speckle noise) که معمولا در تصاویری راداری یا فراصوتی دیده می‌شود و همچنین نویز دانه‌ای فیلم (film grain noise) که در فیلم‌های آنالوگ وجود دارد از این نوع نویز هستند. فیلتر میانگین غیرمحلی (non-local means filter) که کل تصویر را به پیکسل‌هایی در دسته‌های کوچک تقسیم می‌کند و با استفاده از شباهت این دسته‌ها نویز را تشخیص می‌دهد و حذف می‌کند یک روش از بین بردن نویز ضرب‌شونده مخصوصا نویز نقطه‌ای است.

سوال ۳) د)

وقتی به صورت رندوم پیکسل‌های کاملا روشن ۲۵۵ یا کاملا تاریک ۰ در تصویر رخ بدهد، نویز نمک‌فلفل داریم. دلیل آن می‌تواند پیکسل‌های معیوب در سنسور یا مشکل انتقال دیجیتالی داده باشد. برای حذف این نویز می‌توان از فیلتر میانه استفاده کرد زیرا باعث از بین رفتن داده‌های پرت مانند ۰ و ۲۵۵ می‌شود. همچنین می‌توان از alpha-trimmed mean filter استفاده کرد که درصد مشخصی از داده‌های بسیار زیاد و بسیار کم را حذف می‌کند.

سوال ۴)

چهارشنبه
آبان
۲ ربیع الاول ۱۴۴۳
3 Nov 202

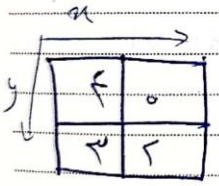
سؤال ۴ الف) جمع مقادیر سطرهای تبدیل فوریه همان نقطه (۰,۰) تبدیل فوریه است

$$F(u,v)=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\,e^{-j2\pi\left(ux/M+vy/N\right)}$$

$$\xrightarrow[v=0]{u=0} F(0,0)=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\,e^{0}$$

جمع مقادیر تصویر

سؤال ۴) ب)



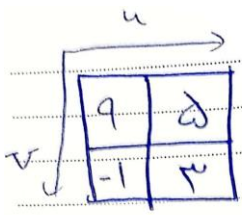
$u=0, v=0 \Rightarrow F(0,0)$ طبق الف ۹

$u=0, v=1 \Rightarrow F(0,1)=4\times e^{-j2\pi(0)}+0+2\times e^{-j2\pi(\frac{1}{2})}+2\times e^{-j2\pi(\frac{1}{2})}=-1$

$u=1, v=0 \Rightarrow F(1,0)=4+0+3\times e^{0}+2\times e^{-j2\pi(\frac{1}{2})}=5$

$u=1, v=1 \Rightarrow F(1,1)=4+0+3\times e^{-j2\pi(0+\frac{1}{2})}+2\times e^{-j2\pi(\frac{1}{2}+\frac{1}{2})}$

$=3$



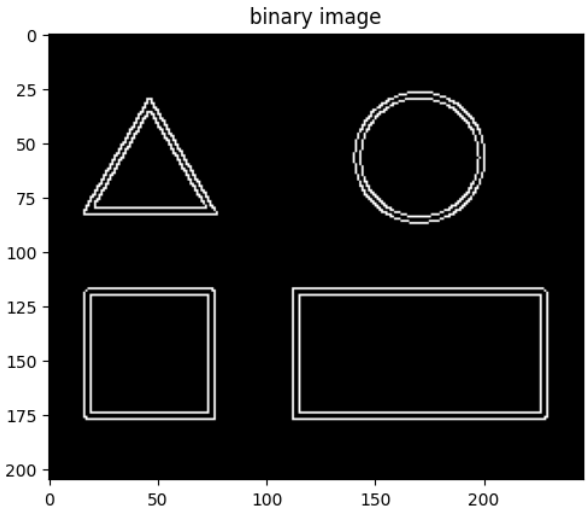
تبدیل فوریه ۱

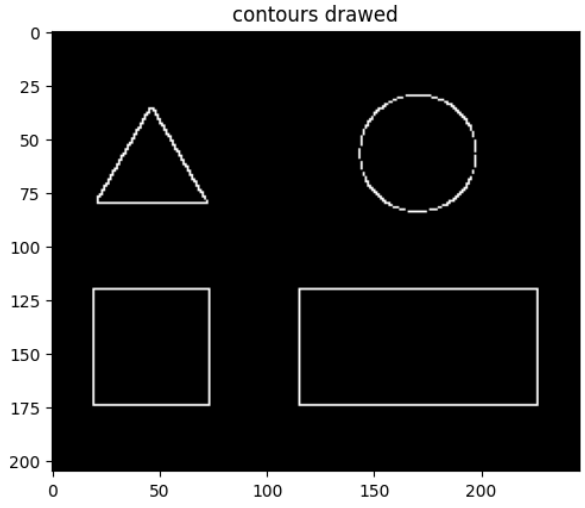
سوال ۵) الف)

تصویر را می‌خوانیم.

سوال ۵) ب)

کانتور انحنایی تشکیل شده از نقاط به‌هم‌پیوسته است که رنگ یا شدت روشنایی مشابهی دارند. تابع cv2.findContours کانتور ها با استفاده از یک نسخه بهینه‌شده الگوریتم سوزوکی پیدا می‌کند. پارامتر ورودی اول این تابع تصویر ورودی است که باید باینری باشد پس از canny استفاده می‌کنیم. پارامتر دوم نشان می‌دهد چه کانتورهایی را می‌خواهیم که در اینجا cv2.RETR_EXTERNAL خطوط خارجی اشکال را بازمی‌گرداند. پارامتر سوم cv2.CHAIN_APPROX_SIMPLE است که بدین معنی می‌باشد که نقاط کمتر و بهینه‌ای را بازمیگرداند مثلا به جای تمام نقاط خط، دو نقطه ابتدا و انتهای خط را برمی‌گرداند. برعکس cv2.CHAIN_APPROX_NONE که تمام نقاط را در کانتور بازمی‌گرداند. در نهایت با استفاده از cv2.drawContours کانتورهای یافت‌شده را رسم می‌کنیم. پارامترهای این تابع به ترتیب عکس ورودی، آرایه‌ای کانتورها، ایندکس کانتوری که قرار است رسم شود (منفی یک یعنی همه کانتورها)، رنگ و شدت می‌باشند.

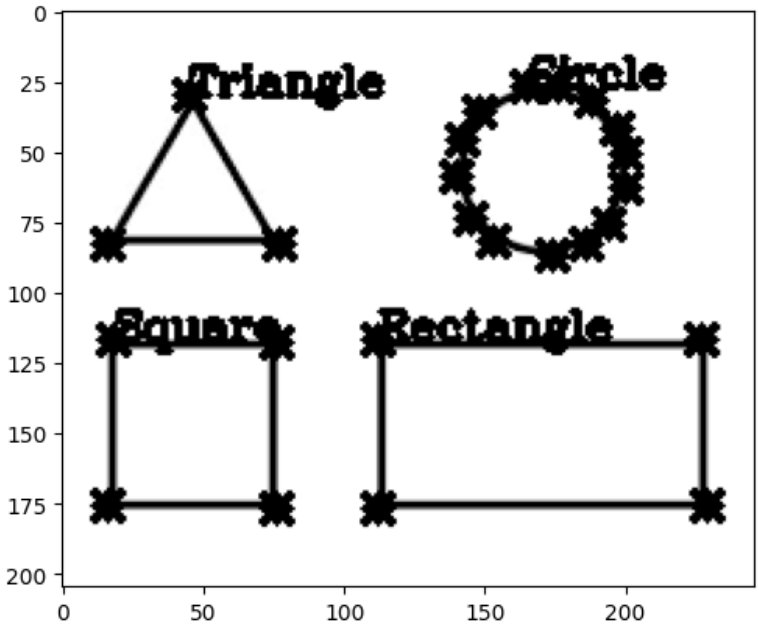




سوال ۵) ج)

تابع cv2.approxPolyDP کانتور و اپسیلون و closed را ورودی می‌گیرد و یک شکل تخمینی که به اندازه اپسیلون به کانتور ورودی شبیه است را برمی‌گرداند. Closed نیز یک بولین است که مشخص می‌کند کانتور یک منحنی بسته است یا باز. برای به دست آوردن اپسیلون از cv2.arclength استفاده کردیم که طول/محیط منحنی را می‌دهد.

برای هر کانتور تخمین می‌زنیم، نقاط موجود در تخمین را با مارکر علامت می‌زنیم و با توجه به تعداد نقاطی که در هر تخمین بازگردانده شده، شکل اشیا را تشخیص می‌دهیم. باید توجه داشت مستطیل و ربع هر دو چهارنقطه تخمینی دارند اما اگر طول مساوی عرض باشد یعنی مربع است.



سوال ۶) الف)

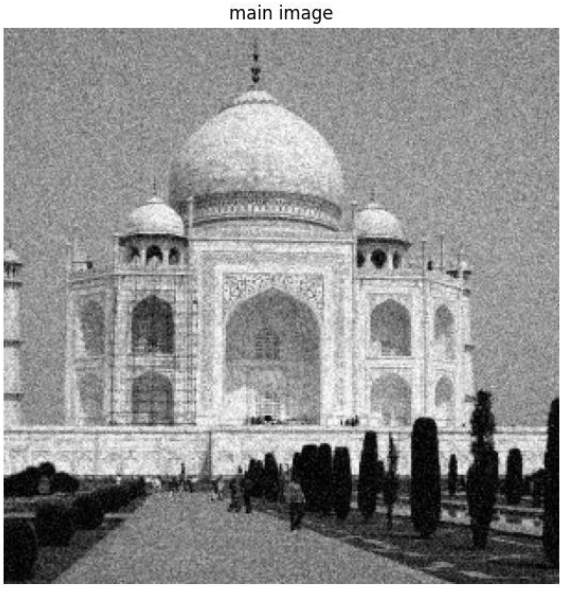
برای reflect101 از np.pad با mode='reflect' استفاده می‌کنیم.

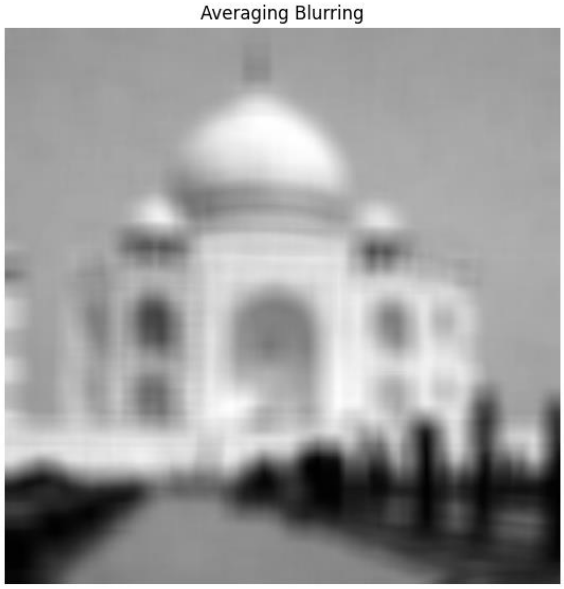
در فیلتر میانگین از پیکسل مربوطه و همسایه‌هایش (با توجه به سایز فیلتر) میانگین می‌گیریم.

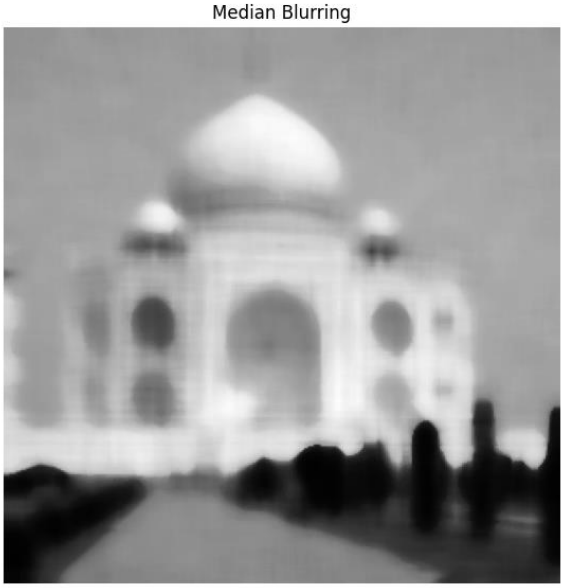
در فیلتر میانه از بین پیکسل مربوطه و همسایه‌هایش میانه را مشخص می‌کنیم و جای آن پیکسل می‌گذاریم.

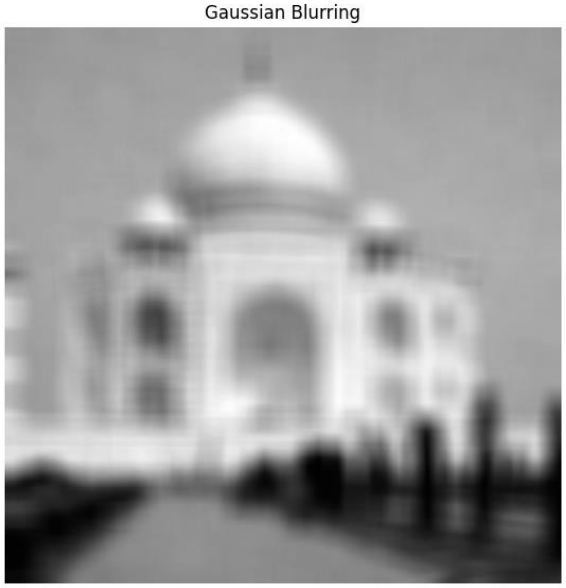
در فیلتر گاوسی نیز همانند سوال یک عمل می‌کنیم. باید بدانیم فیلتر گاوسی در واقع یک نوع میانگین وزن دار است، هرچقدر پیکسل‌ها به پیکسل مربوطه نزدیکتر باشند، وزن بیشتری دارند.

خروجی‌ها بدین صورت می‌باشد:









همچنین با افزایش سایز فیلتر تصویر محوتر و smooth تر می‌شود. مثلا اینجا برای فیلتر میانگین اندازه ۶۰ داده شده است:

سوال ۶) ج)

در اینجا از توابع آماده cv2 استفاده می‌کنیم و می‌بینیم نتایج بسیار مشابهی به دست می‌آید: