

دانشگاه صنعتی امیرکبیر  
( پلی تکنیک تهران )

تمرین سری سه

درس بینایی ماشین

فرهاد دلیرانی

۹۶۱۳۱۱۲۵

[dalirani@aut.ac.ir](mailto:dalirani@aut.ac.ir)

[dalirani.1373@gmail.com](mailto:dalirani.1373@gmail.com)

فهرست

ابزارهای استفاده شده ..... ۱

تمرین ۱ ..... ۲

قسمت یک ..... ۲

قسمت دو ..... ۳

تمرین ۲ ..... ۷

تمرین ۳ ..... ۱۲

ابزارهای استفاده شده

زبان برنامه نویسی: پایتون ۳,۶

محیط توسعه: PyCharm



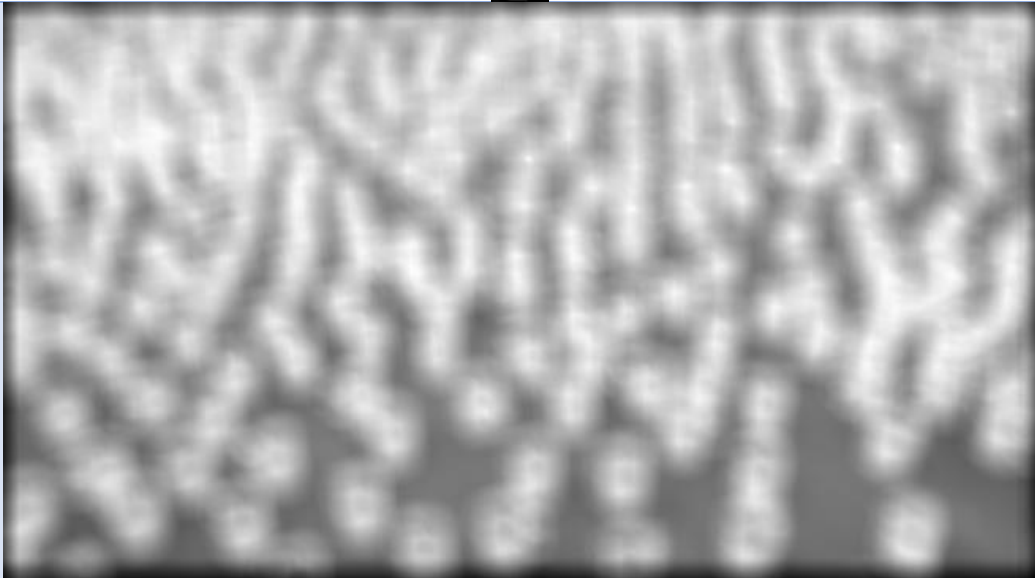

سیستم عامل: Windows 10


## تمرین ۱

کدهای این سوال در problem-1.py قرار دارد.

### قسمت یک



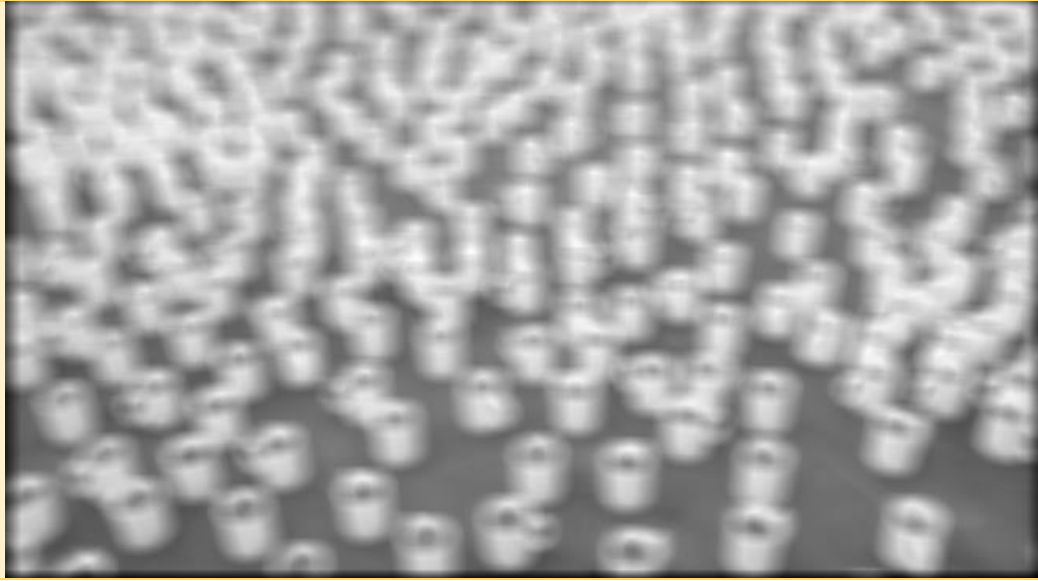
در قسمت تصویر و کلیشه را باهم **correlation** می‌کنیم. در تصویر حاصل از **correlation** شدت روشنایی نقاطی که در آن لیوان وجود دارد بیشتر از سایر نقاط است. ولی برای نمایش بهتر از حاصل **correlation** آستانه‌گیری می‌کنیم و نقاطی که از آستانه کمتر هستند را برابر با صفر قرار می‌دهیم و نقاطی که بزرگ‌تر مساوی آستانه هستند را با نقاط تصویر ورودی جایگزین می‌کنیم. در شکل زیر خروجی‌های این کار را مشاهده می‌کنید:

			تصویر ورودی
			کلیشه
			حاصل <b>Correlation</b> تصویر و کلیشه
			آستانه‌گیری از حاصل <b>Correlation</b>

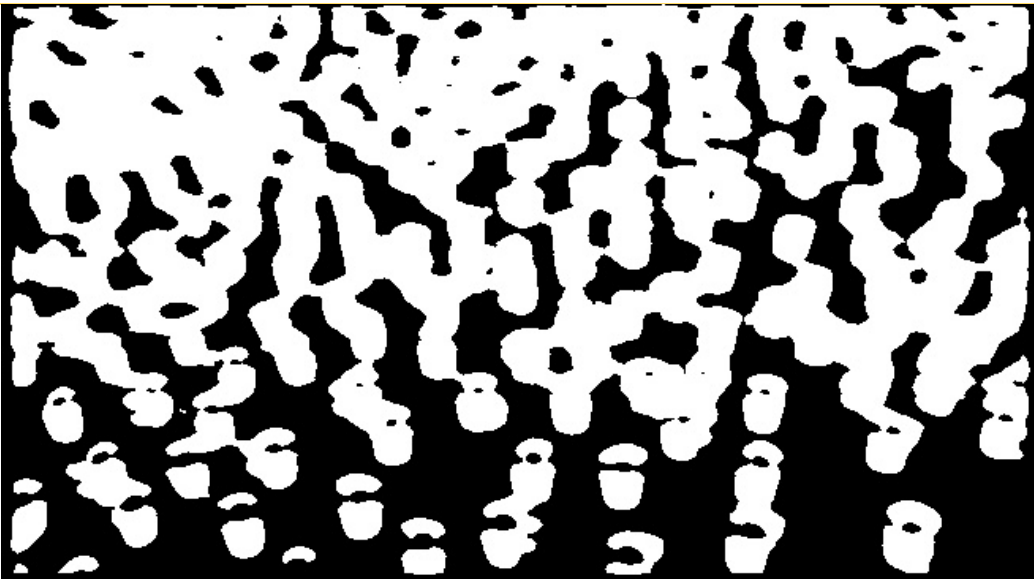
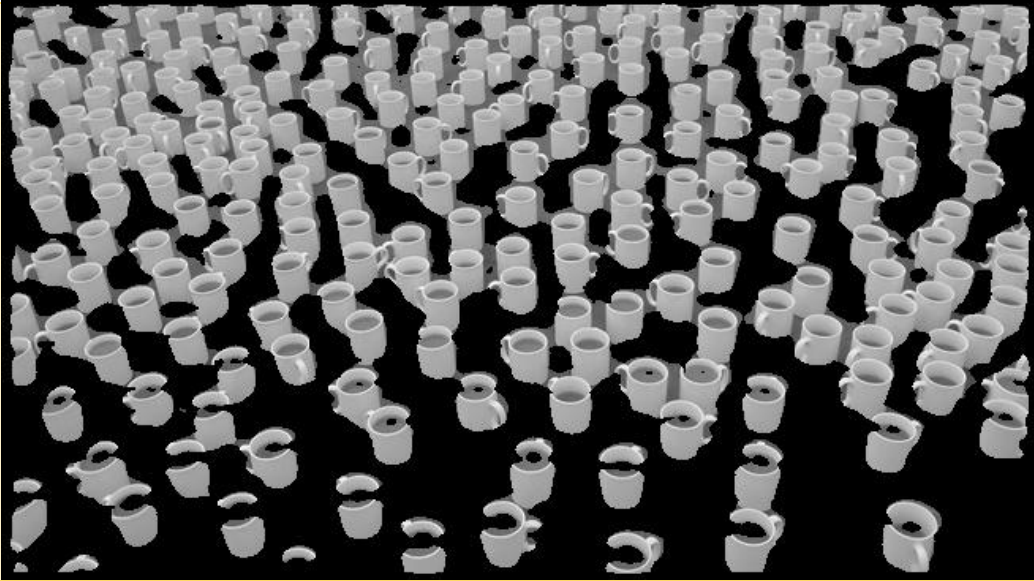
		<p>نقاطی در تصویر که در آستانه‌گیری از حاصل <b>Correlation</b> مقدارشان بیشتر از آستانه بوده است</p>
--	--	--



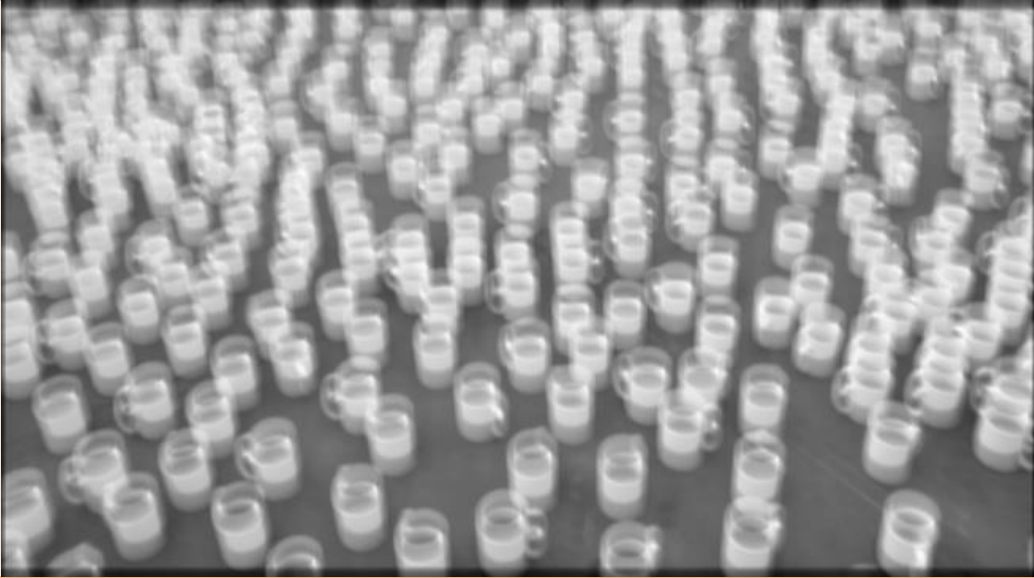
قسمت دو

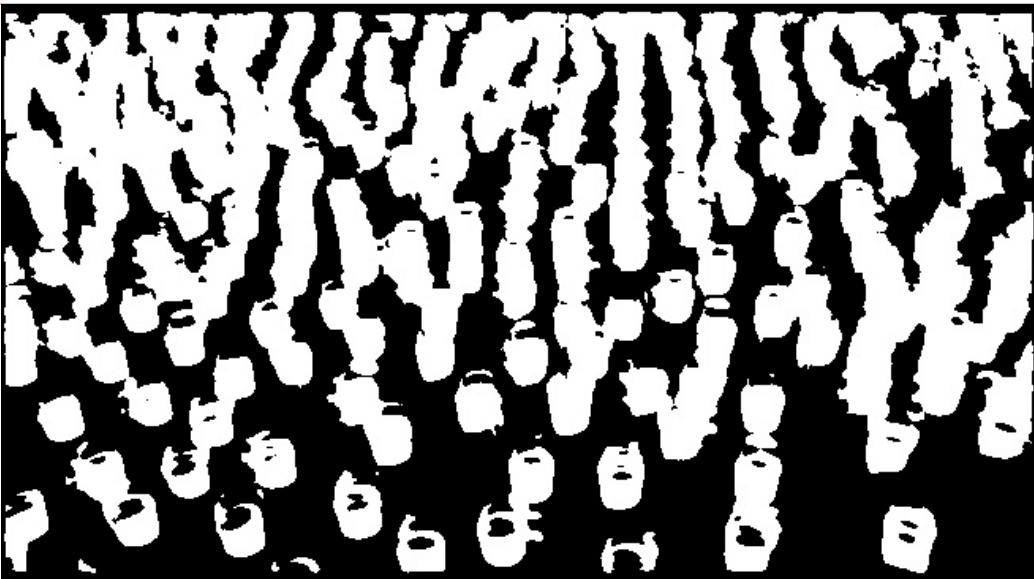
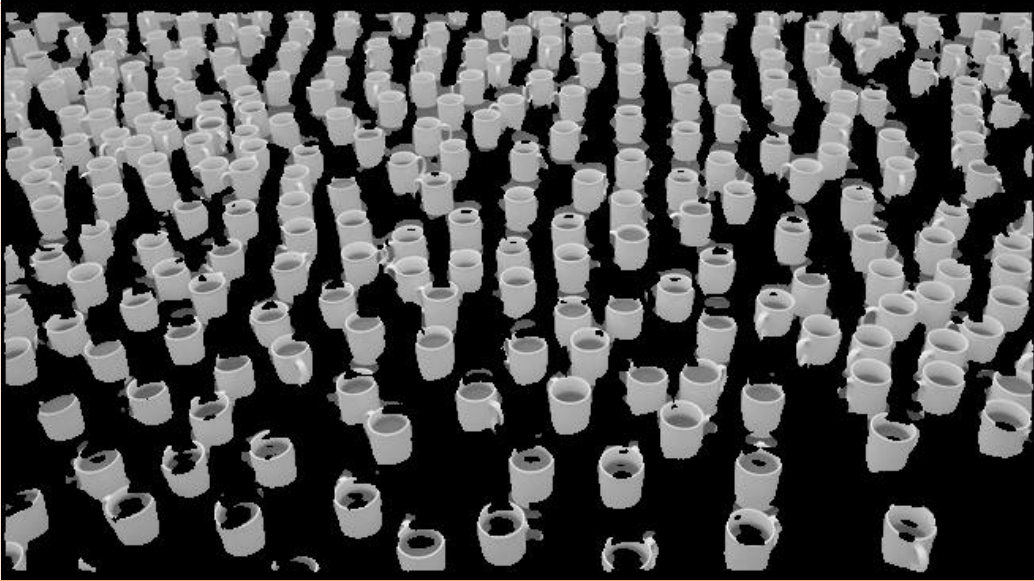
در این قسمت از کلیشه داده شده قسمت‌های متفاوتی را جدا و بررسی می‌کنیم و عملکرد آن‌ها را بر پیدا کردن لیوان‌ها را مشاهده می‌کنیم.

		<p>تصویر ورودی</p>
	<p>قسمت بالایی لیوان</p>	<p>کلیشه</p>
		<p>حاصل <b>Correlation</b> تصویر و کلیشه</p>



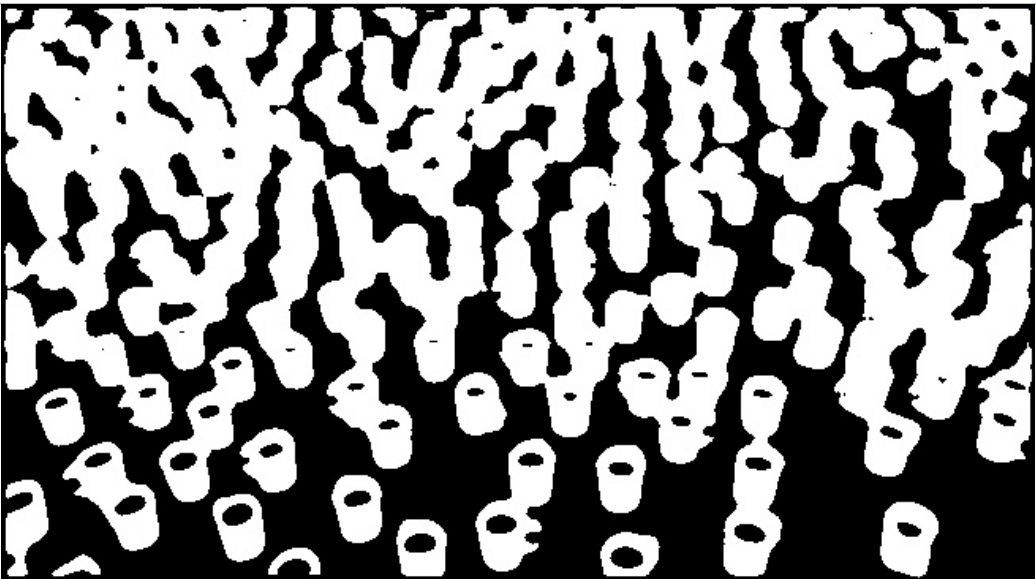

			آستانه‌گیری از حاصل <b>Correlation</b>
			نقاطی در تصویر که در آستانه‌گیری از حاصل <b>Correlation</b> مقدارشان بیشتر از آستانه بوده است

			تصویر ورودی
	<div>  </div> <div> قسمت دسته‌ی لیوان </div>		کلیشه
			حاصل <b>Correlation</b> تصویر و کلیشه

		آستانه‌گیری از حاصل <b>Correlation</b>
		نقاطی در تصویر که در آستانه‌گیری از حاصل <b>Correlation</b> مقدارشان بیشتر از آستانه بوده است

			تصویر ورودی
<div> <div></div> <div>قسمت سفید و میانه لیوان</div> </div>			کلیشه
			حاصل <b>Correlation</b> تصویر و کلیشه



		آستانه‌گیری از حاصل <b>Correlation</b>
		نقاطی در تصویر که در آستانه‌گیری از حاصل <b>Correlation</b> مقدارشان بیشتر از آستانه بوده است



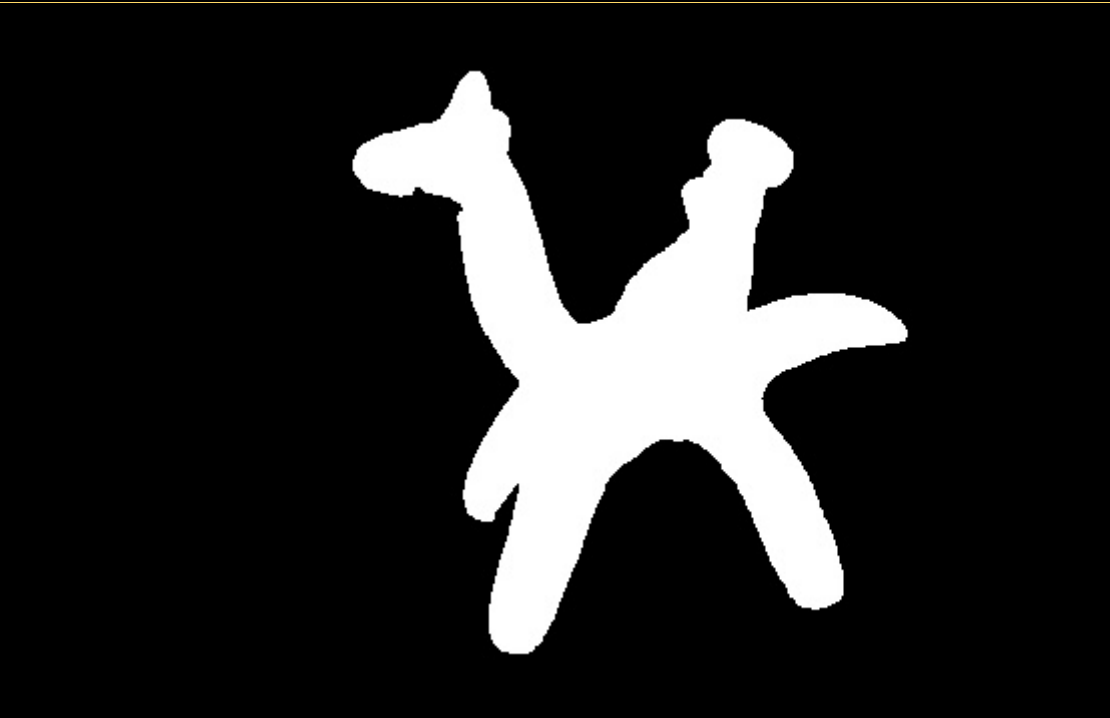
بهترین عملکرد را به ترتیب از راست به چپ این کلیشه‌ها داشته‌اند: قسمت سفید و میانی لیوان، دسته لیوان، قسمت بالایی لیوان که شکلی بیضی مانند دارد و قهوه در آن مشخص است و در آخر کل کلیشه ورودی. کلیشه داده شده یکی از حالت‌هایی است که لیوان در تصویر قرار دارد، لیوان‌های دیگر با توجه به دوری و نزدیکی از دوربین اندازه‌شان متفاوت است، نورپردازی بر روی لیوان‌ها یکسان نیست، لیوان‌ها با زاویه‌های مختلفی قرار گرفته‌اند، کافی درون بعضی از آن‌ها مشخص نیست و مهم تر از همه بعضی لیوان‌ها با هم دیگر هم‌پوشانی دارند. همه این‌ها سبب شده است که کلیشه داده شده نسبت به کلیشه‌های دیگر که خودمان ایجاد کرده‌ایم عملکرد خوبی نداشته باشد. قسمت بالایی لیوان که حالت بیضی مانند دارد و قهوه در آن دیده می‌شود، عملکرد بهتری از کلیشه ورودی داده شده دارد. یکی از دلایلی که این کلیشه بهتر عمل کرده است این است که هم‌پوشانی در قسمت بالایی لیوان نیست. قسمت سفید و میانه لیوان بهترین عملکرد را داشته است، زیرا تنها شی‌های سفید در صحنه لیوان‌ها هستند، به همین دلیل این کلیشه بهترین عملکرد را داشته است.

با توجه به آزمایش‌های انجام داده شده، دو نتیجه می‌گیریم. یک: استفاده از کلیشه‌ای کوچک‌تر که از کلیشه اصلی جدا شده است سرعت را بیشتر می‌کند زیرا محاسبات کمتری باید انجام شود. دو: نه تنها استفاده از کلیشه‌ای که از کلیشه‌ی اصلی جدا کرده‌ایم سرعت را افزایش می‌دهد بلکه می‌تواند منجر به نتایج بهتر و دقیق‌تری شود.

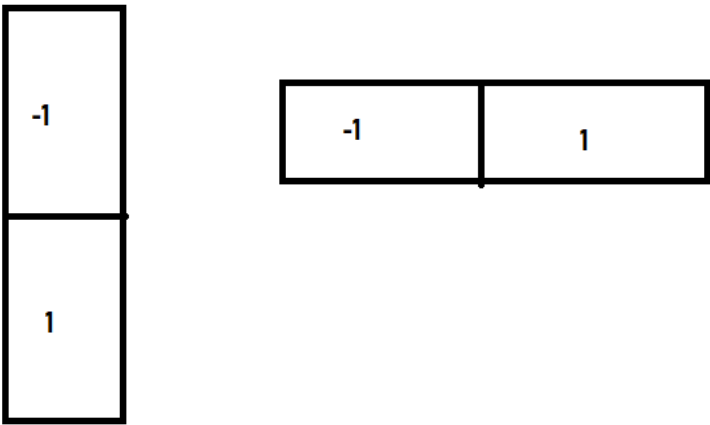


تمرین ۲

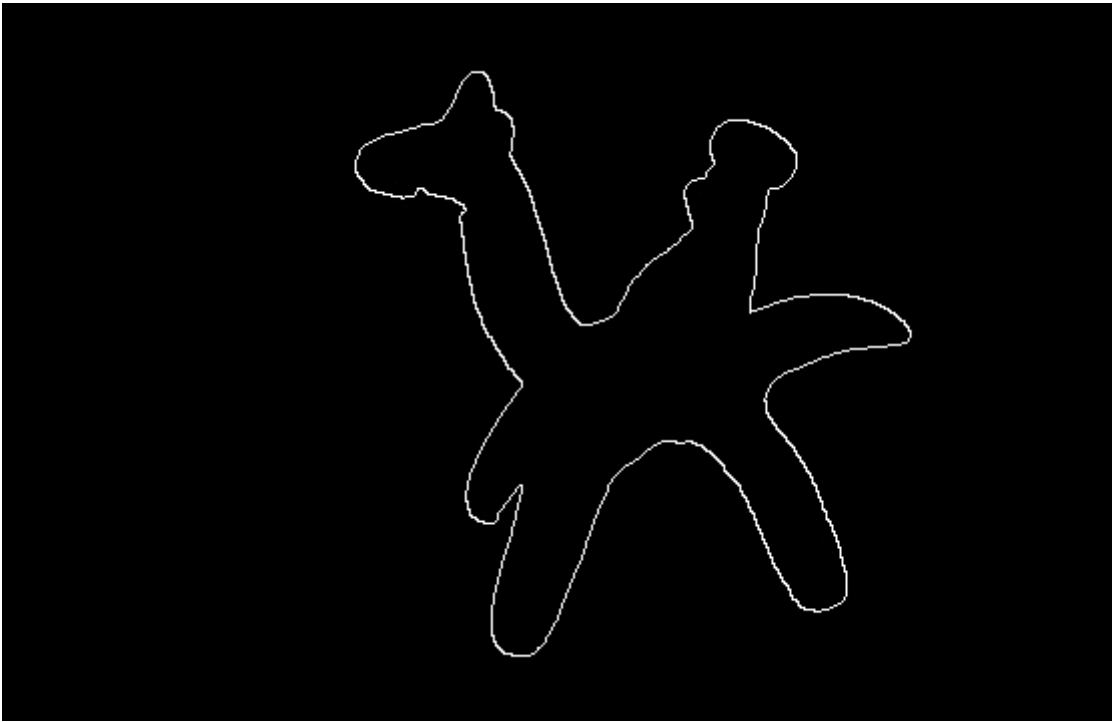
کدهای این تمرین در `problem-2.py` قرار دارد. در ابتدا باید شی را از تصویر جدا کنیم، زیر اگر به صورت مستقیم لبه‌یابی انجام دهیم علاوه بر لبه‌های بیرونی شکل، لبه‌هایی را پیدا می‌کنیم که درون شکل هستند. برای پیدا کردن لبه‌های دور شکل ابتدا تصویر را با یک فیلتر  $5 \times 5$  که تمام المنت‌های آن برابر با یک بر روی بیست و پنج هستند، کانولوشن می‌کنیم. سپس عمل `Thresholding` را انجام می‌دهیم با استفاده از آستانه برابر با ۵ شی را از پس‌زمینه جدا می‌کنیم. در تصویرهای زیر این کارها را مشاهده می‌کنید:

			تصویر ورودی
			
			تصویر بلور شده
			جدا کردن شی از پس زمینه با استفاده از آستانه گیری

اکنون که شی و پس‌زمینه را با استفاده از آستانه‌گیری از هم جدا کرده‌ایم. اقدام به پیدا کردن لبه‌ها می‌کنیم. لبه با ضخامت یک کار ایجاد کد freeman را آسان‌تر می‌کند، برای اینکه لبه‌هایی به ضخامت یک از دو فیلتر زیر استفاده می‌کنیم:



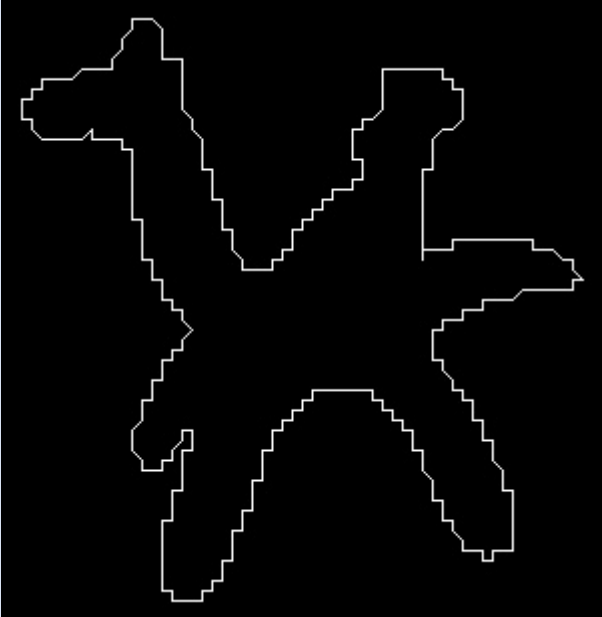
با اعمال فیلترهای بالا، لبه‌های شکل را به دست می‌آورید که در شکل زیر آن‌ها را مشاهده می‌کنید:



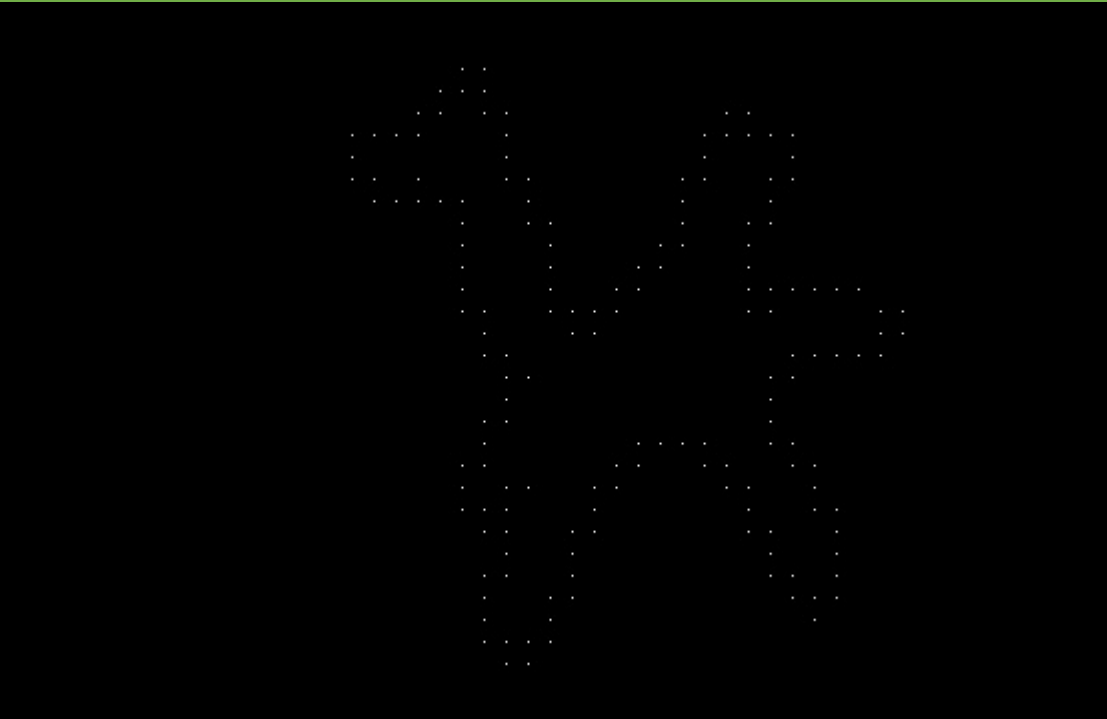
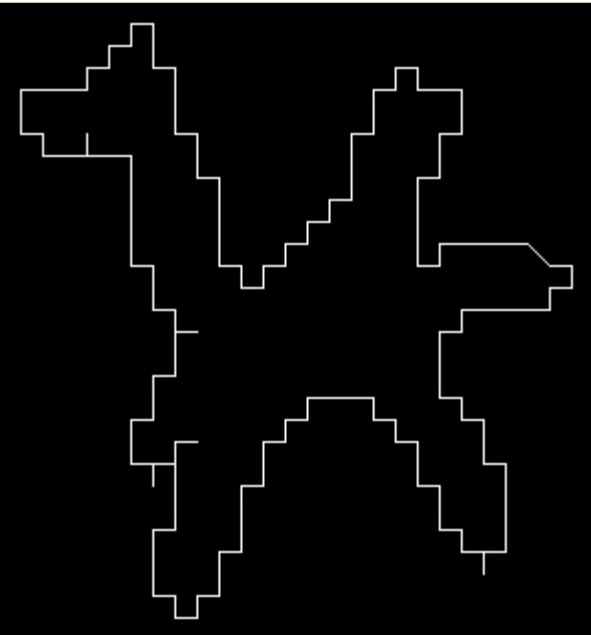
اکنون لبه‌ها را داریم، با استفاده از الگوریتم Boundary Following که در صفحه‌ی ۷۹۶ ویرایش ۳ گنزالس آمده است، پیکسل‌های روی مرز را به ترتیب چرخش ساعت به دست می‌آوریم. سپس یک Grid بر روی مرز قرار می‌دهیم. بر اساس آن کد Freeman code را به دست می‌آوریم و کد به دست آمده را رسم می‌کنیم. این کار را برای سه Grid متفاوت انجام داده‌ایم. در زیر خروجی‌ها را مشاهده می‌کنید.

در Gird اول بین هر دو پیکسل روی Grid در تصویر اصلی ۴ پیکسل فاصله وجود دارد:

			حاصل قرار دادن grid بر روی مرز بیرونی شی. فاصله هر دو نقطه‌ی رو گرید ۴ پیکسل است
[0, 0, 0, 0, 0, 0, 0, 0, 4, 6, 0, 0, 2, 6, 0, 0, 0, 0, 0, 0, 6, 2, 0, 7, 1, 6, 1, 5, 2, 6, 0, 0, 0, 0, 1, 7, 0, 0, 2, 6, 0, 2, 6, 0, 2, 6, 2, 0, 0, 2, 6, 1, 7, 2, 6, 2, 6, 0, 2, 6, 0, 2, 6, 0, 1, 7, 0, 2, 6, 0, 0, 0, 0, 6, 0, 2, 6, 6, 2, 0, 6, 1, 7, 2, 6, 0, 2, 6, 0, 1, 7, 0, 2, 6, 0, 2, 6, 2, 6, 2, 6, 2, 6, 2, 0, 0, 0, 0, 0, 2, 6, 2, 6, 2, 6, 2, 6, 2, 0, 6, 2, 0, 0, 6, 2, 0, 0, 6, 2, 0, 6, 2, 0, 6, 2, 6, 2, 6, 0, 0, 6, 2, 6, 0, 0, 0, 0, 0, 6, 2, 0, 0, 6,			کد فریمن مستقل از

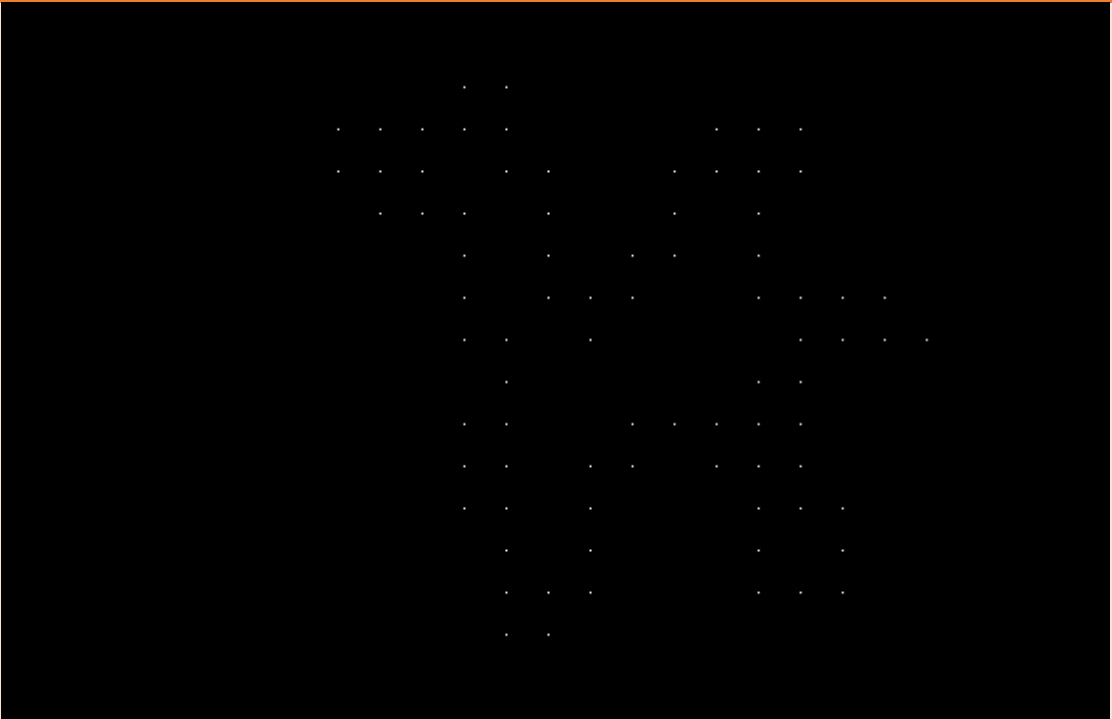
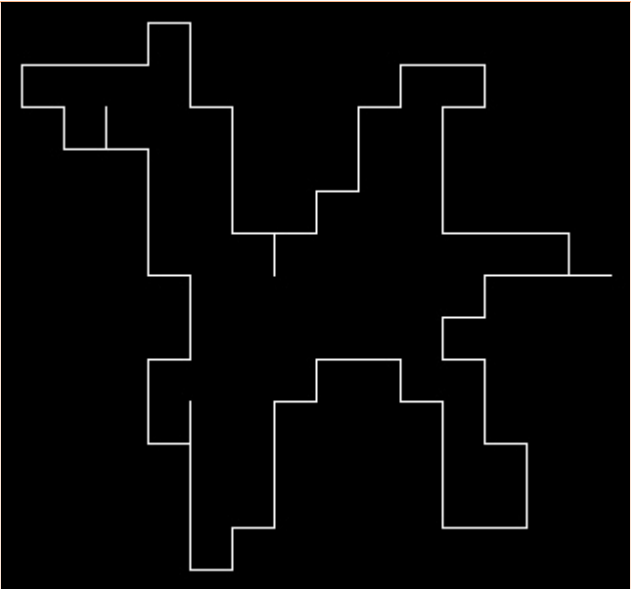
2, 0, 0, 0, 6, 2, 0, 2, 2, 7, 1, 6, 2, 6, 0, 6, 1, 7, 0, 7, 1, 0, 6, 2, 0, 6, 2, 0, 6, 2, 6, 2, 7, 2, 7, 2, 6, 2, 6, 0, 2, 6, 0, 2, 6, 0, 0, 0, 2, 6, 0, 0, 0, 0, 0, 0, 2, 6, 2, 0, 0, 6, 3, 7, 0, 0, 0, 7, 7, 2, 6, 0, 6, 2, 6, 2, 6, 0, 0, 1, 7, 0, 0, 2, 7, 1, 7, 1, 6, 0, 7, 7, 0, 0, 2, 0, 6, 0, 0, 0, 0, 1, 7, 1, 7, 0, 0, 2, 6, 0, 0, 2, 6, 0, 0, 2, 6, 0, 1, 7, 2, 0, 0, 2, 6, 2, 6, 2, 0, 6, 2, 6, 2, 6, 2, 6, 2, 6, 0, 2, 6, 2, 0, 2, 6, 0, 0, 6, 2, 6, 1, 1, 0, 0, 0, 6, 0, 0, 0, 0, 0, 6, 2, 6, 2, 6, 2, 6, 0, 0, 7, 7, 1, 1, 0, 0, 6, 2]			نقطه‌ی شروع و چرخش
			رسم کد فریمن حاصل

در Gird دوم بین هر دو پیکسل روی Grid در تصویر اصلی ۱۰ پیکسل فاصله وجود دارد:


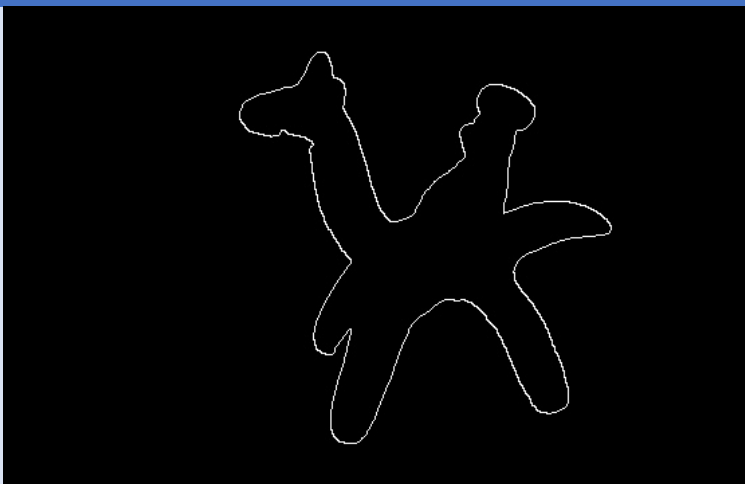
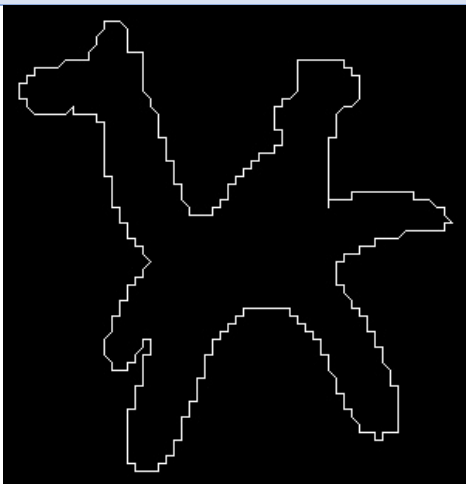
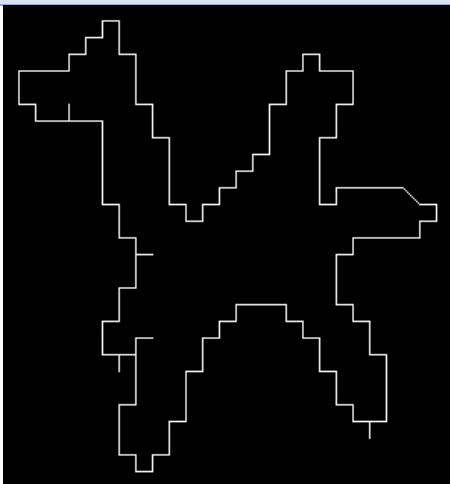
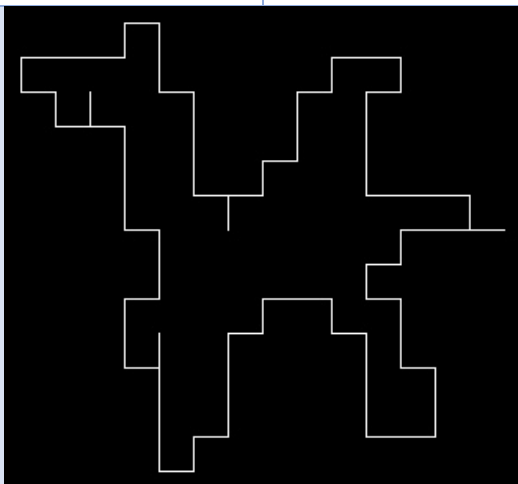
			حاصل قرار دادن grid بر روی مرز بیرونی شی. فاصله هر دو نقطه‌ی رو گرید ۴ پیکسل است
[0, 0, 0, 0, 2, 0, 6, 4, 6, 0, 6, 2, 6, 0, 6, 0, 0, 2, 6, 2, 6, 2, 6, 6, 0, 2, 6, 0, 0, 2, 6, 0, 2, 6, 0, 0, 0, 2, 6, 2, 2, 6, 2, 6, 2, 6, 2, 0, 0, 6, 2, 0, 6, 2, 6, 6, 2, 0, 6, 0, 6, 2, 0, 6, 2, 0, 0, 0, 2, 2, 6, 0, 0, 0, 7, 1, 6, 6, 2, 6, 0, 0, 0, 2, 6, 2, 0, 0, 2, 6, 2, 6, 0, 2, 6, 0, 0, 0, 6, 2, 4, 2, 6, 2, 6, 0, 2, 6, 0, 2, 6, 2, 6, 2, 0, 0, 2, 6, 2, 6, 2, 0, 6, 2, 0, 0, 6, 2, 0, 6, 2, 6, 6, 2, 6, 0, 0, 6, 2, 0, 0, 0, 6, 4, 2, 6, 2, 4, 2, 6, 0, 6, 2, 0, 6, 2, 0, 6, 4, 6, 2, 6, 0, 2, 6]			کد فریمن مستقل از نقطه‌ی شروع و چرخش
			رسم کد فریمن حاصل



در Gird سوم بین هر دو پیکسل روی Grid در تصویر اصلی ۲۰ پیکسل فاصله وجود دارد:

			حاصل قرار دادن grid بر روی مرز بیرونی شی. فاصله هر دو نقطه ی رو گرید ۴ پیکسل است
			
[0, 0, 0, 4, 6, 6, 0, 6, 2, 0, 2, 6, 0, 0, 2, 6, 4, 6, 6, 2, 6, 6, 0, 0, 2, 6, 6, 0, 2, 6, 0, 0, 2, 6, 4, 6, 2, 6, 2, 0, 6, 2, 6, 0, 6, 6, 2, 0, 0, 2, 0, 0, 6, 2, 4, 0, 0, 2, 6, 2, 2, 6, 0, 2, 6, 0, 6, 0, 6, 0, 0, 2, 6, 2, 0, 2, 6, 2, 0, 0, 6, 2, 6, 6]			کد فریمن مستقل از نقطه ی شروع و چرخش
			رسم کد فریمن حاصل

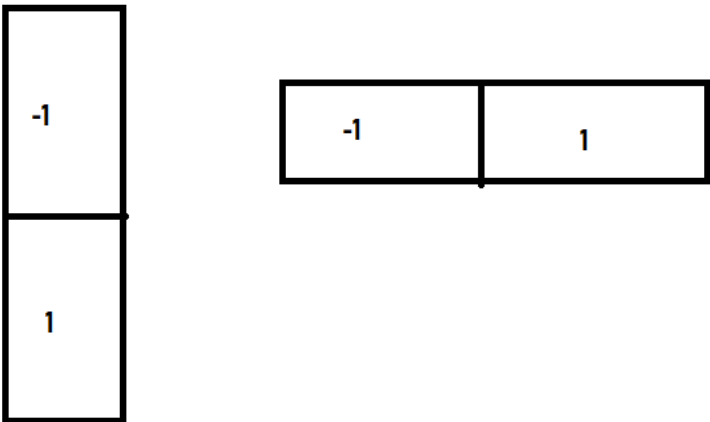
برای مقایسه بهتر کد فریمن رسم شده‌ی حاصل از سه Grid مختلف را در زیر رسم می‌کنیم:

	
تصویر ورودی	لبه بیرونی شی
	
Grid با فاصله پیکسلی ۴	Grid با فاصله پیکسلی ۱۰
	
Grid با فاصله پیکسلی ۲۰	

هر چه Grid بزرگ‌تری استفاده کرده‌ایم کد فریمن حاصل کوتاه‌تر شده است، همین‌طور با مقایسه‌ی مرز شی با کدهای فریمن مختلف متوجه می‌شویم هر چه Grid بزرگ‌تر شده است، جزئیات کمتری از مرز شی نمایش داده می‌شود و کد فریمن از مرز شی دور می‌شود و کد فریمن به دست آمده نسبت به تغییرات جزئی رباست‌تر می‌شود. در حالی که هر چه Grid کوچک‌تر شده است، کد فریمن رسم شده بیشتر شبیه مرز بیرونی شی شده است و جزئیات بیشتری را نشان می‌دهد و نسبت به نویز و تغییرات جزئی حساس‌تر می‌شود.

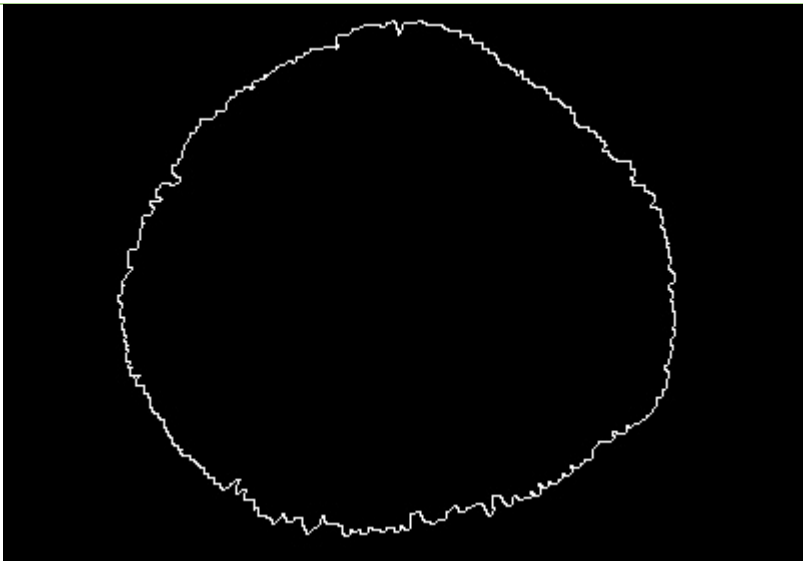
تمرین ۳

کدهای این قسمت در `problem-3.py` قرار دارد. قبل از ساخت `Fourier Transform` باید مرزهای خارجی شی را استخراج کنیم. برای این کار مانند سوال قبل عمل می‌کنیم، ابتدا تصویر را با یک فیلتر  $3 \times 3$  که تمام المنت‌های آن برابر با یک تقسیم بر نه است، کانولوشن می‌کنیم در ادامه با استفاده از `Thresholding` شی را از پس‌زمینه جدا می‌کنیم و بعد با استفاده از فیلترهای زیر لبه‌های بیرونی شی به ضخامت یک را استخراج می‌کنیم

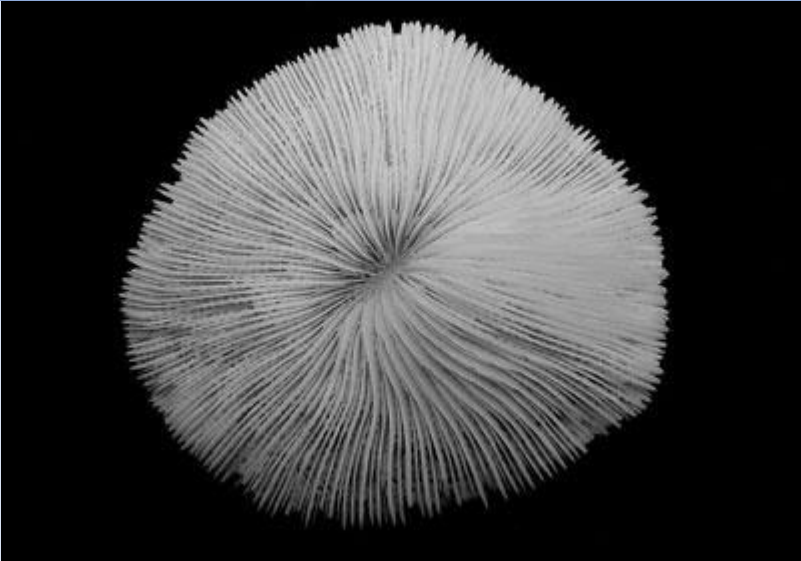
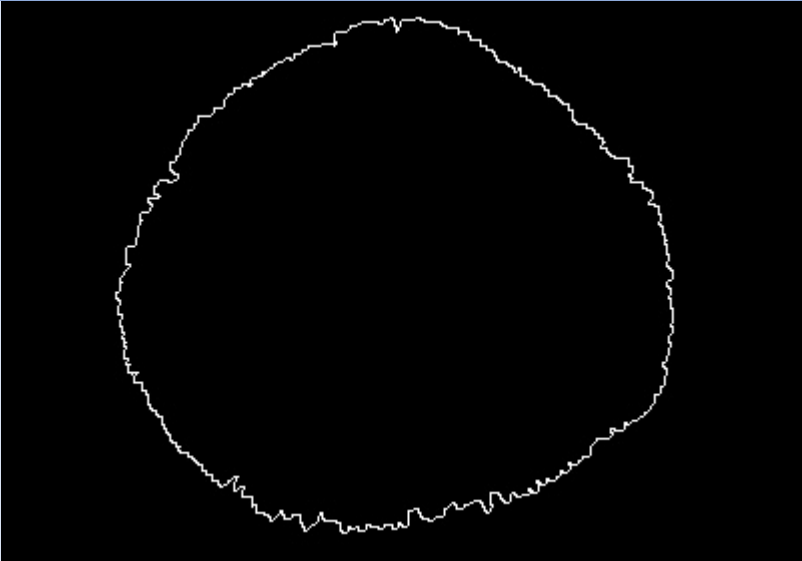
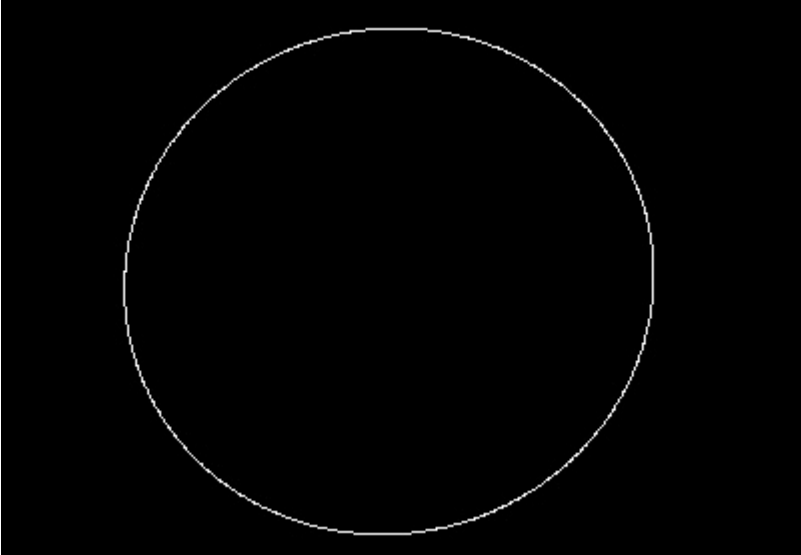
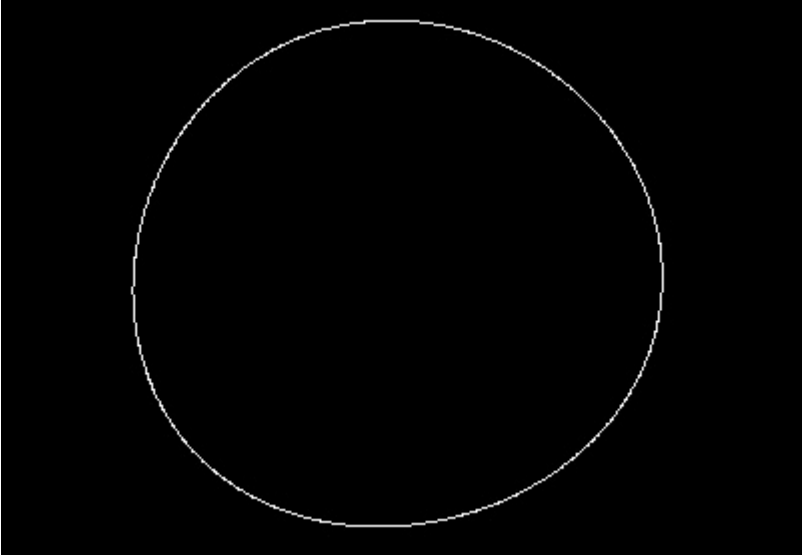
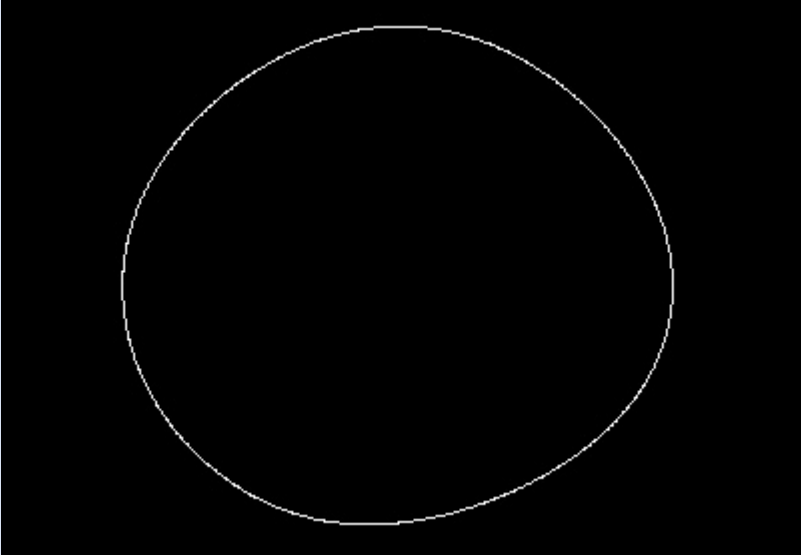
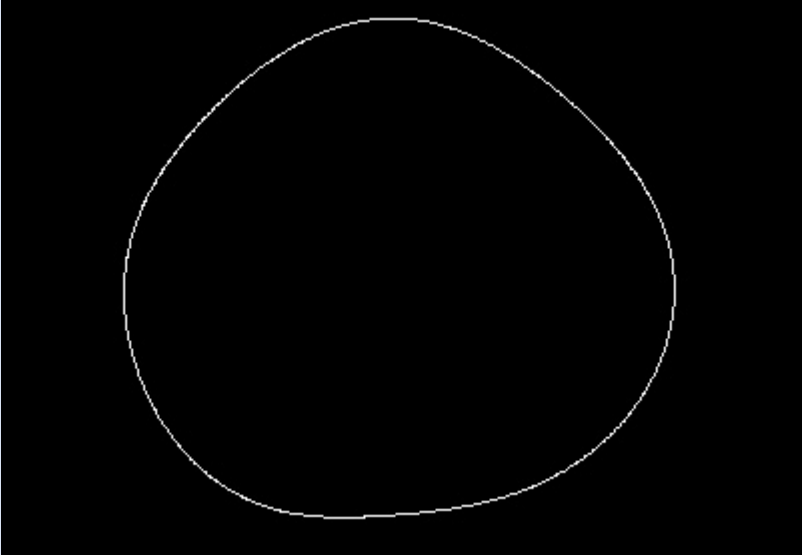


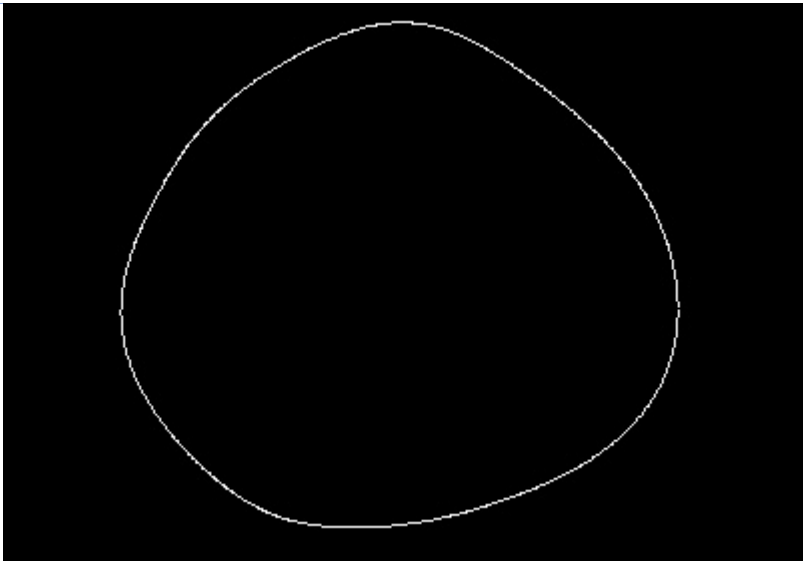
			تصویر ورودی
			تصویر بلور شده
			جدا کردن شی از پس زمینه با استفاده از آستانه گیری



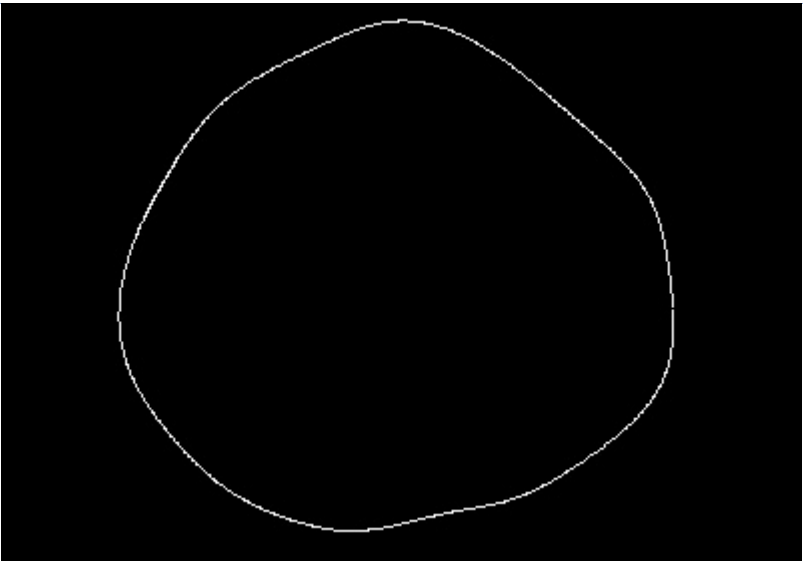
			لبه‌های خارجی شی
--	--	--	---------------------

بعد از پیدا کردن لبه‌های خارجی شی، با استفاده از الگوریتم Boundary Following که در صفحه ۷۹۶ ویرایش ۳ کتاب گنزالس شرح داده شده است، نقطه‌های روی مرز را به ترتیب ساعتگرد پیدا می‌کنیم. سپس نقطه‌ها را که به صورت  $(x,y)$  هستند را به صورت یک عدد **complex** به صورت  $x+iy$  در می‌آوریم. سپس از تبدیل فوریه‌ی یک بعدی استفاده می‌کنیم و معادل دنباله در حوزه‌ی فوریه را به دست می‌آوریم. سپس  $p$  عضو با فرکانس کمتر را نگه می‌داریم و باقی ضرایب فوریه را برابر صفر قرار می‌دهیم و با استفاده از تبدیل معکوس فوریه گسسته دنباله اولیه را دوباره باز سازی می‌کنیم. به این ترتیب توصیف‌گر فوریه را ایجاد می‌کنیم. در زیر توصیفگرهای مختلف فوریه به ازای  $p$  های مختلف را مشاهده می‌کنید:

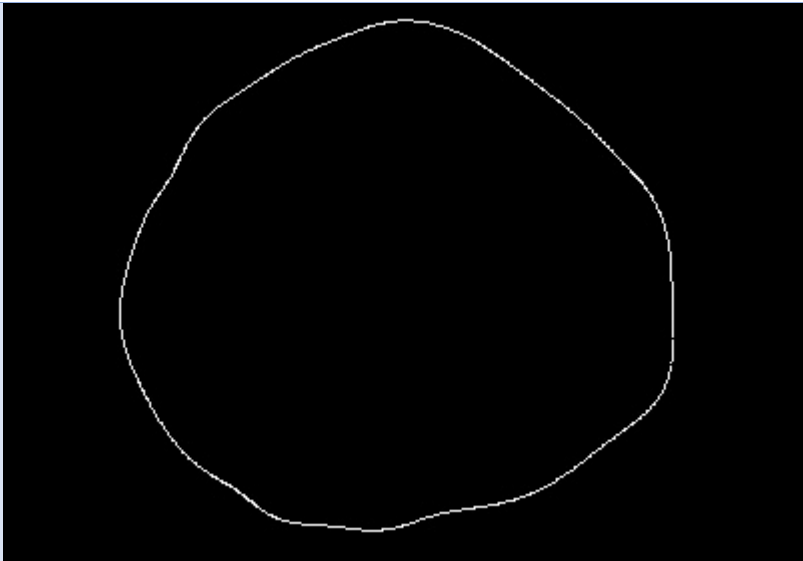
 <p>تصویر ورودی</p>	 <p>مرزهای خارجی شکل</p>
 <p><math>P = 2</math></p>	 <p><math>P = 4</math></p>
 <p><math>P = 6</math></p>	 <p><math>P = 8</math></p>



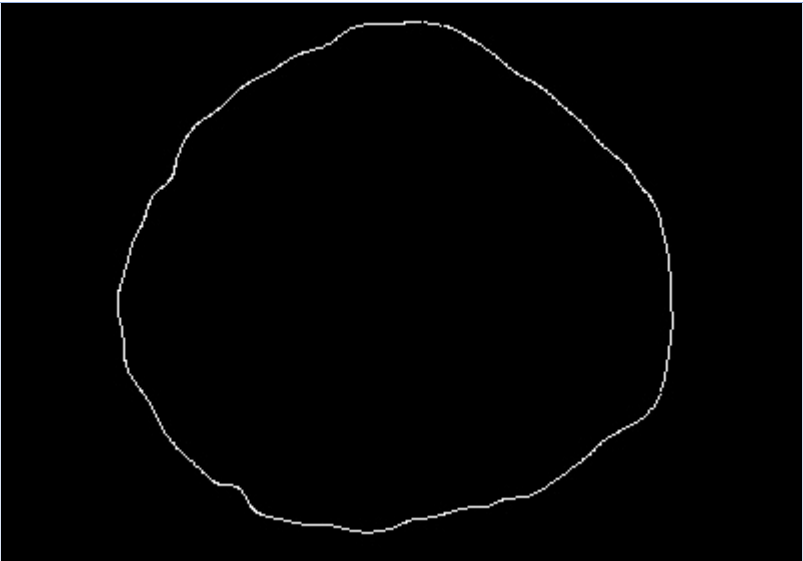
P = 12



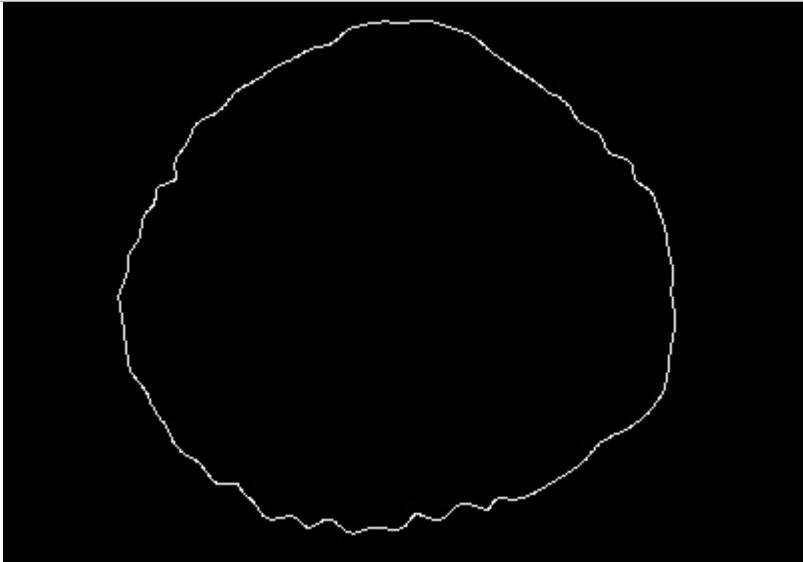
P = 18



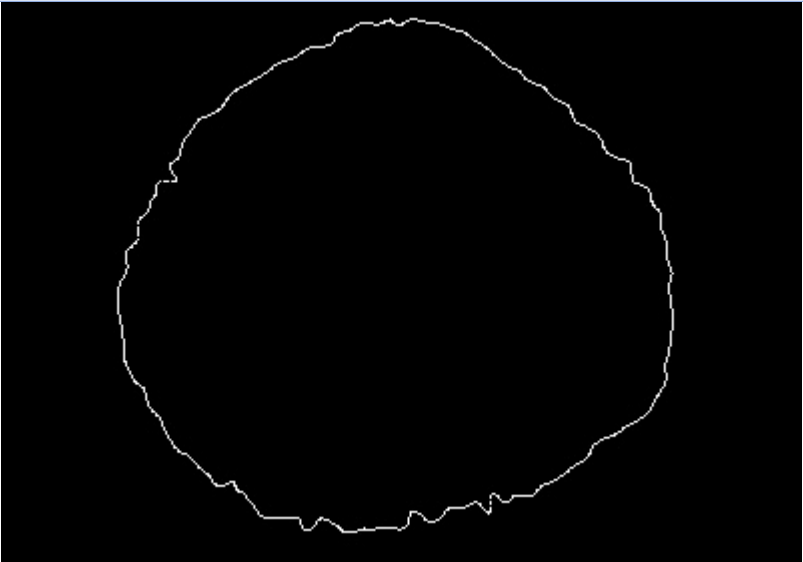
P = 30



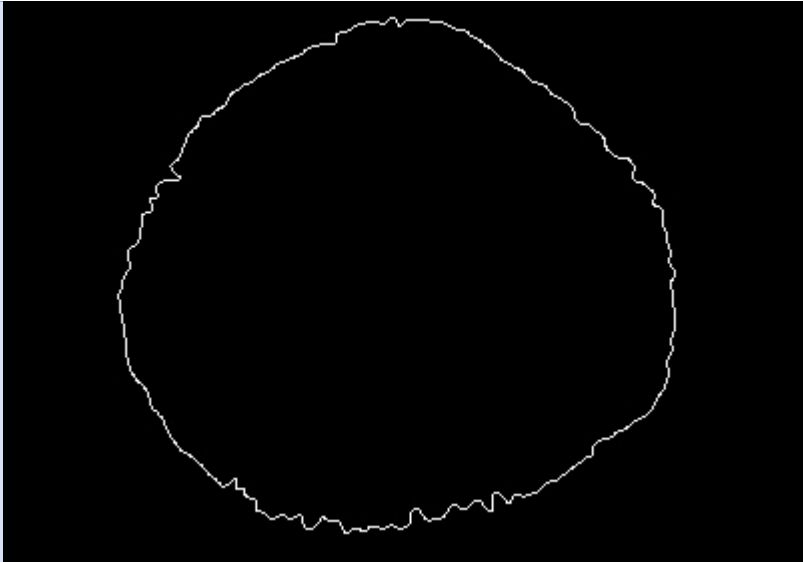
P = 60



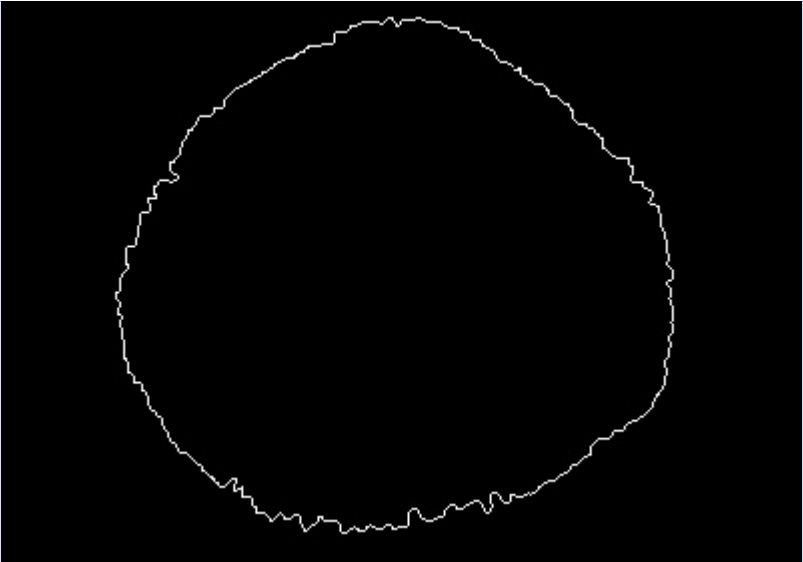
P = 90



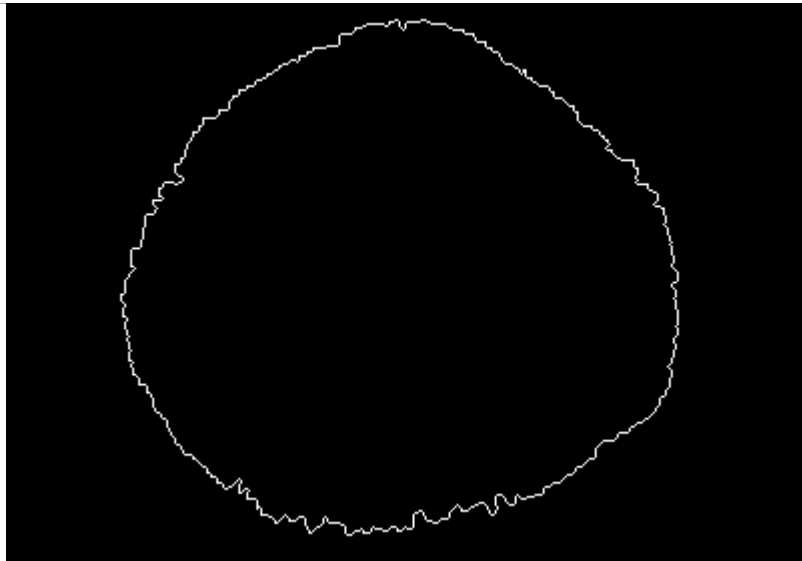
P = 140



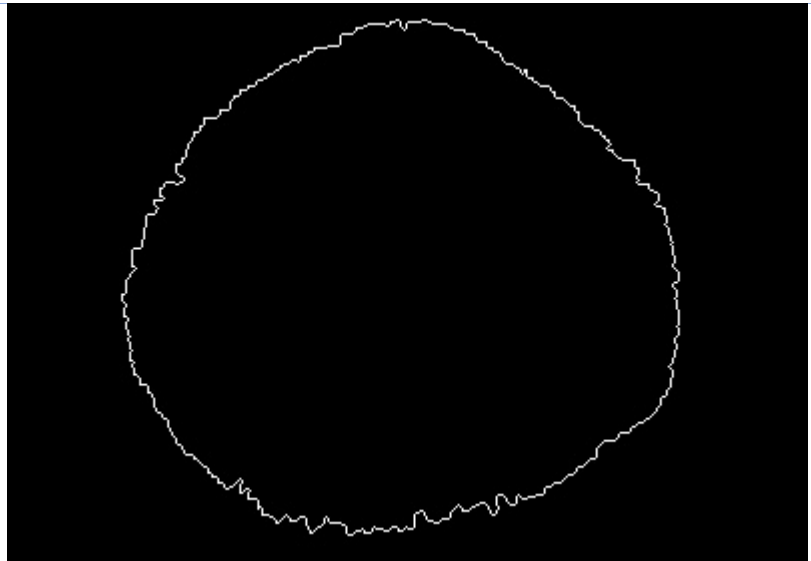
P = 200



P = 300



**P = 700**



**P = 930**

همانطور که در نتایج دیده می‌شود، هر چه  $P$  افزایش یافته است دنباله‌های بیشتری در حوزه‌ی فوریه را نگه داشته‌ایم در نتیجه هنگام انجام تبدیل فوریه معکوس دنباله‌ای که به دست می‌آید به دنباله‌ی اولیه و قبل از اعمال تبدیل فوریه نزدیک تر است. وقتی  $p$  برابر با دو است شکل بازسازی شده یک دایره است، هر چه  $p$  افزایش می‌یابد شکل از حالت گردی خارج می‌شود و دندانه‌های تیزتر و بیشتری در اطراف شکل ایجاد می‌شود. هر چه  $p$  بیشتر باشد توصیفگر به دست آمده به نویز و تغییرات جزئی بیشتر حساس می‌شود.