دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

گروه مستقل مهندسی رباتیک

تمرین دوم درس بینایی ماشین

استاد درس:

دکتر صفابخش

تدریس‌یار:

مهندس مجد

نام دانشجو:

نوید خزاعی

۹۲۱۳۵۰۰۸

فروردین ۱۳۹۳

فهرست مطالب

۱ بخش اول: لبه‌یابی

۱.۱ Sobel - Nevatia-Babu

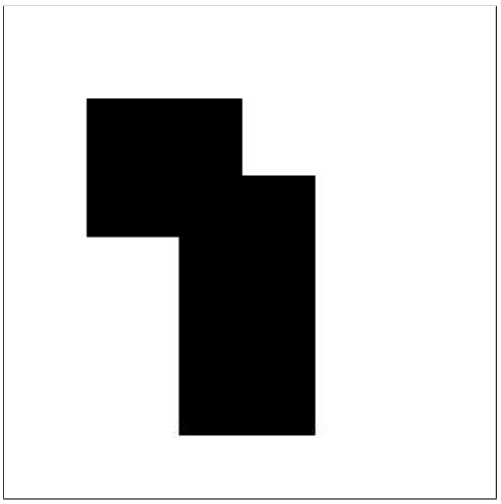
پرسش: لبه‌یاب سوبل و نواتیا ببو را باهم مقایسه کرده و مزایا و معایبشان را به طور خلاصه ذکر کنید. نتیجه‌ی اعمال این دو لبه‌یاب را بر تصویر دل‌خواه نشان دهید. فیلترهای نواتیا ببو در ادامه آمده‌است.

پاسخ: لبه‌یاب سوبل از فیلترهای زیر برای محاسبه‌ی گرادیان (مشتق شدت روشنایی)، استفاده می‌کند.

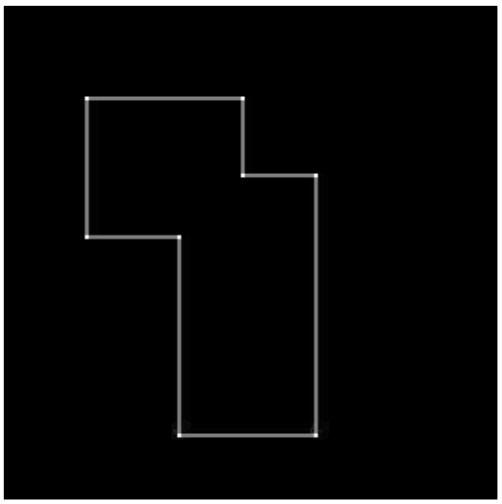
$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$$

برای یافتن لبه‌ها، لازم است هر یک از فیلترهای بالا را بر تصویر اعمال کنیم (همگشت یا Convolution کنیم) و سپس تصاویر حاصل را با هم جمع کنیم. نتایج اعمال آن بر دو تصویر در زیر آمده‌است. در این پیاده‌سازی، توسط تابع filter2D ، هر کدام از فیلترهای بالا را اعمال نمودیم و نتایج حاصل را، با استفاده از تابع addWeighted ، با وزن‌های یکسان 0.5 ، جمع نمودیم.

برای تصویر خانه، نتایج اعمال سوبل در هر یک از راستاهای عمودی و افقی نیز آورده شده‌است.



تصویر اصلی



اعمال Sobel

تصویر اصلی

اعمال سوبل

سوبل در راستای x

سوبل در راستای y

در روش نواتیا ببو، همان‌گونه که در صورت تمرین آمده‌است، از شش فیلتر در زوایای مختلف صفر، ۳۰، ۶۰، ۹۰، ۱۲۰ و ۱۵۰ درجه، استفاده شده‌است. این باعث می‌شود تا گوشه‌های جهت‌داری که از دید سوبل پنهان می‌ماند یا تضعیف می‌شد نیز به خوبی مشخص باشد. ایراد این روش، آن است که باید پس از اعمال فیلتر، یک عمل Thresholding (آستانه‌ای کردن) انجام شود و در صورت نیاز، نازک‌سازی نیز لازم است. در ادامه نمونه‌ای از اعمال این فیلترها آورده شده‌است. نتایج اعمال هر شش فیلتر، با وزن‌های مساوی با هم جمع شده‌اند.

اعمال فیلترهای نواتیا ببو

اعمال فیلترهای نواتیا ببو

از دیگر معایب نواتیا ببو، کند بودن آن به دلیل زیاد بودن عملیات آن است. از طرفی، فیلتر سوبل، در عین سریع‌بودن و عدم نیاز به عملیات اضافی، لبه‌های جهت‌دار ضعیف را از دست می‌دهد.

Canny ۲.۱

پرسش: به طور خلاصه توضیح دهید لبه‌یاب کنی چگونه عمل می‌کند. هدف از کرنل گوسی در این لبه‌یاب چیست؟ این لبه‌یاب را بر تصویر شماره ۴ اعمال کنید.

پاسخ: این لبه‌یاب که به لبه‌یاب بهینه نیز معروف است، سه ویژگی اصلی دارد:

- **خطای کم:** فقط لبه‌های موجود را به خوبی تشخیص می‌دهد.
- **مکان‌یابی خوب:** فاصله‌ی لبه‌ی تشخیص داده‌شده تا لبه‌ی واقعی در تصویر، کمینه است.
- **واکنش کمینه:** به هر لبه فقط یک‌بار واکنش نشان می‌دهد.

نحوه کار این لبه‌یاب، به این شکل است که ابتدا توسط یک کرنل گاسی، نویز را از تصویر حذف می‌کند. سپس با استفاده از کرنل سوبل، که در زیر آورده شده‌است، گرادیان شدت روشنایی را حساب می‌کند.

سپس جهت و بزرگی گردایان را از روابط زیر به‌دست می‌آورد.

$$G = \sqrt{G_x^2 + G_y^2} \qquad (۱)$$

$$\theta = arctan(\frac{G_y}{G_x}) \quad (۲)$$

که جهت گرادیان، به یکی از زوایای صفر، ۴۵، ۹۰ و یا ۱۳۵ درجه، گرد می‌شود. سپس از پیکسل‌هایی که بیشینه‌ی محلی نیستند صرف نظر می‌کند، زیرا احتمال این که این نقاط بر روی لبه باشند، پایین است. پس از آن، در مرحله‌ی پایانی، از دو مقدار آستانه برای تشخیص لبه استفاده می‌کند؛ به این ترتیب که اگر در یک پیکسل مقدار گرادیان بیشتر از حد آستانه‌ی بالا باشد، آن پیکسل به عنوان لبه انتخاب می‌شود. اگر مقدار گرادیان از حد آستانه‌ی پایینی کمتر باشد، انتخاب نمی‌شود. و در نهایت، اگر مقدار گرادیان مقداری بین دو حد آستانه باشد، تنها در صورتی انتخاب می‌شود که به یک پیکسل روی لبه (بالاتر از حد آستانه) متصل باشد. در این روش، پیشنهاد می‌شود نسبت حد آستانه‌ی بالا به پایین، بین ۲:۱ و ۳:۱ باشد.

همان‌گونه که گفته‌شد، کرنل گاسی برای رفع نویز استفاده می‌شود، چون کرنل سوبل که برای پیدا کردن لبه‌ها استفاده می‌شود، بسیار به نویز حساس است. در OpenCV برای اعمال این لبه‌یاب، ابتدا با فراخوانی تابع زیر، عمل رفع نویز را با یک کرنل گاسی 3×3 انجام می‌دهیم:

```
blur( src, dst, Size(3,3));
```

سپس تابع زیر را برای اعمال کنی با کرنل 3×3 فراخوانی می‌کنیم:

```
Canny( dst, edgesCanny, lowThreshold, highThreshold, 3);
```

در زیر، نتیجه‌ی اعمال این روش بر روی تصویر ۴ آمده‌است. حد آستانه‌ی بالا و پایین تصویر مشخص شده‌است.

تصویر ۴

کنی با آستانه‌ی پایین ۱ و آستانه‌ی بالای ۲۵۵

۳.۱ پارامترهای Canny

پرسش: لبه‌یاب کنی سه پارامتر دارد، تغییر مقدار این پارامترها چه تاثیری بر نتیجه‌ی حاصله دارد؟ اندازه‌ی کرنل ۳، ۵ و ۷ را بر تصویری دل‌خواه اعمال کرده و نتیجه را مقایسه کنید. اگر آستانه‌ی بالا را از ۲۵۵ به ۱۲۸ تغییر دهیم چه تغییری بر تصویر نهایی ایجاد می‌شود؟ نتیجه‌ی تغییر آستانه‌ی پایین از ۱ به ۲۲۰ چیست؟

پاسخ:

- **حد آستانه‌ی پایین:** هر اندازه کمتر باشد، لبه‌های ضعیف‌تر کم‌تری حذف می‌شوند و لبه‌های ضعیف بیشتری، برای بررسی بعدی انتخاب می‌شوند. زیاد کردن این مقدار نیز باعث شکسته‌تر شدن لبه‌های یافت‌شده می‌شود.
- **حد آستانه‌ی بالا:** هر اندازه بیشتر باشد، تنها لبه‌های قوی‌تر انتخاب خواهند شد.
- **اندازه کرنل:** از آنجا که کرنل بزرگ‌تر، گرادیان را با توجه به محدوده‌ی وسیع‌تری محاسبه می‌کند، از دقت بیشتری برخوردار است.

نتیجه‌ی لبه‌یابی و اعمال تغییرات بر روی عکس خانه در بخش قبلی، در زیر آورده شده‌است.

کنی ۲۵۵-۱ با کرنل ۳

کنی ۱۲۸-۱ با کرنل ۳

کنی ۲۵۵-۱ با کرنل ۳

کنی ۲۵۵-۲۲۰ با کرنل ۳

همان‌گونه که مشاهده می‌کنید، با کاهش حد آستانه‌ی بالا به ۱۲۸، لبه‌های ضعیف‌تر بیشتری یافت شده‌اند. تشخیص تمامی برگ‌ها و چمن‌ها حاصل این کاهش است. همچنین با افزایش حد آستانه‌ی پایین، لبه‌های ضعیف مانند چمن‌ها حذف شده‌اند و لبه‌های قوی مانند پیرامون خانه و ساعت در گوشه‌ی پایین تصویر، بیشتر به چشم می‌آیند.

نتیجه‌ی تغییرات اندازه کرنل نیز آورده شده‌است. مشاهده می‌کنید که با افزایش اندازه، دقت بیشتر شده‌است.

کنی ۲۵۵-۱ با کرنل ۵

کنی ۲۵۵-۱ با کرنل ۷

۴.۱ مقایسه کنی و لبه‌یاب‌های کلاسیک

پرسش: لبه‌یاب کنی نسبت به لبه‌یاب‌های کلاسیک (مثل سوبل و نواتیا ببو) چه مزایا و معایبی دارد؟ نتیجه این دو روش را مقایسه کنید.

پاسخ: در استفاده از کنی، پیدا کردن مقادیر آستانه‌ای مناسب برای هر تصویر متفاوت و زمان‌بر است. همچنین اجرای خود الگوریتم از لبه‌یاب‌های کلاسیک زمان‌برتر است. با این حال لبه‌های یافت‌شده توسط کنی از دقت و بهینگی بیشتری برخوردار هستند و نازک‌سازی اضافی نیز نیاز نیست. دلایل بهینگی در بخش ۲.۱ آورده‌شد. از طرفی لبه‌یاب‌های کلاسیک، ممکن است لبه‌های مجزای بهتری به دست دهند که با مقایسه‌ی دو عکس زیر در خصوص چمن‌های تشخیص داده‌شده به این نتیجه می‌رسیم. چرا که کنی، ممکن است لبه‌های ضعیف متصل به لبه‌های قوی را، بر روی یک لبه‌ی قوی تشخیص دهد، حال آن‌که در سوبل چنین نیست و لبه‌ها از بزرگی و جهت مستقل از یک‌دیگر برخوردار هستند. همچنین احتمال شکستگی لبه‌ها در کنی وجود دارد ولی در سوبل و نواتیا ببو این چنین نیست.

اعمال سوبل

کنی ۲۵۵-۱ با کرنل ۳

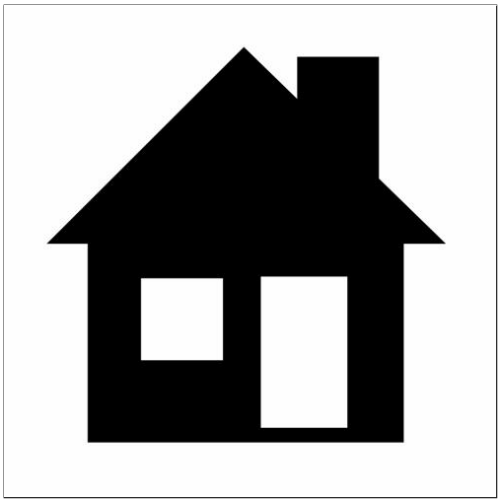
۲ بخش دوم: تبدیل هاف

۱.۲ تبدیل هاف برای تشخیص خط

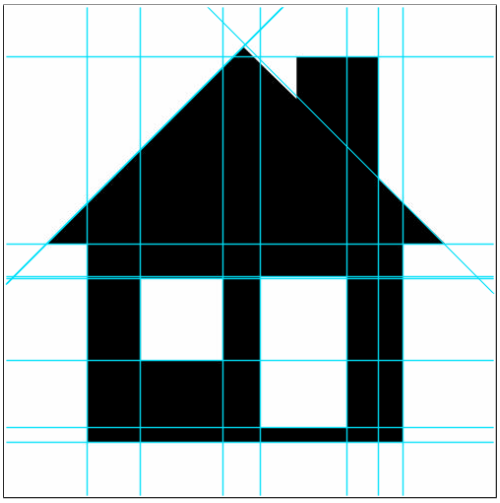
پرسش: یکی از روش‌های تشخیص خط، استفاده از تبدیل هاف است. معمولا ورودی تبدیل هاف چیست؟ تبدیل هاف را بر تصویری شامل خطوط اعمال کرده و خطوط بدست آمده را رسم کنید.

پاسخ: از آن‌جا که تبدیل هاف، نیاز دارد نقاط موجود در عکس را برای تشخیص خط، به فضای پارامتری ببرد، بنابراین تنها نقاط موجود در ورودی هاف باید نقاطی باشند که ممکن است بر روی خطی قرار داشته‌باشند. با این فرض،

ورودی تبدیل هاف باید لبه‌های موجود در عکس باشد تا در صورت وجود لبه‌ای به صورت خط صاف، تشخیص داده‌شود. در زیر نتیجه‌ی اعمال هاف با استفاده از تابع HoughLines آورده شده‌است.



تصویر اصلی

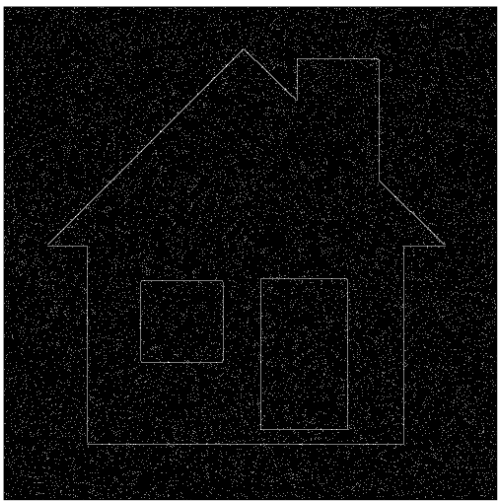


هاف: $\theta = 1^\circ, r = 0.1, threshold = 50$

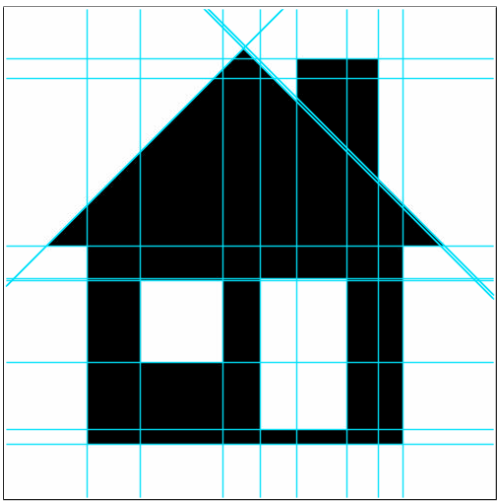
۲.۲ اثر نویز بر هاف

پرسش: نویز چه تاثیری بر تبدیل هاف دارد؟ برای بررسی این سوال، به تصویر حاصله از لبه‌یاب، نویز گوسی اضافه کرده و تبدیل هاف را اعمال کنید. تبدیل هاف تا چه واریانسی از نویز گوسی را می‌تواند تحمل کند و خطوط را تشخیص دهد؟ (تصویر شماره ۵)

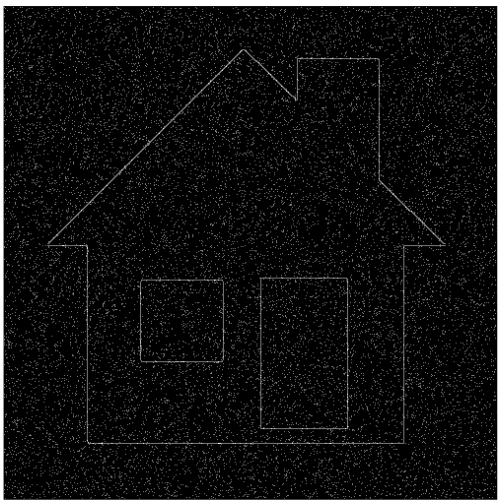
پاسخ: نویز سبب می‌شود هاف نقاطی با فواصل زیاد که در تصویر اصلی روی یک خط نیستند ولی در نویز خام اضافه‌شده، روی یک خط هستند را تاثیر دهد و خطوط اضافه‌ی بسیاری را پیدا کند. در ادامه چند نمونه آورده شده‌است که مشاهده می‌کنیم با میانگین ۳۰۰ برای نویز گاسی، نتیجه‌ی هاف از واریانس ۷۰ به بالا غیر قابل قبول است.



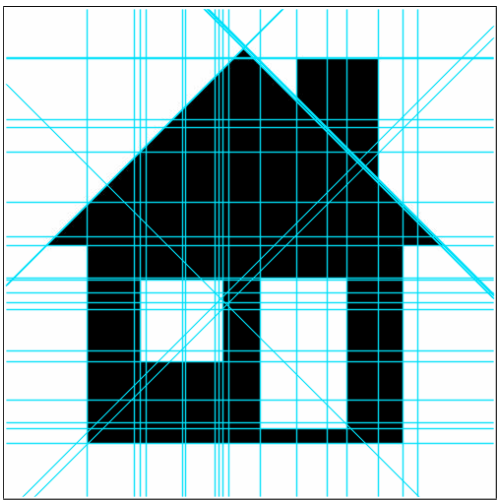
تصویر ورودی $\sigma = 63$



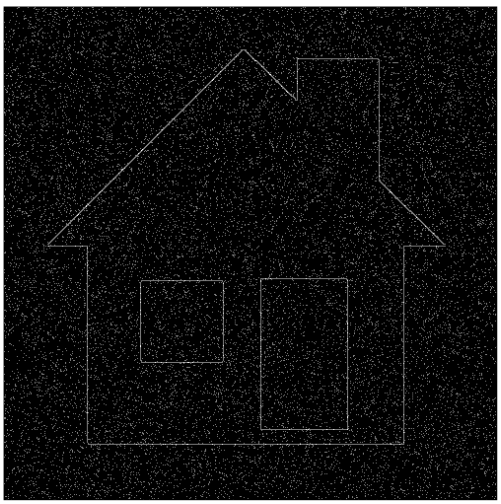
هاف: $\theta = 1^\circ, r = 0.1, threshold = 50$



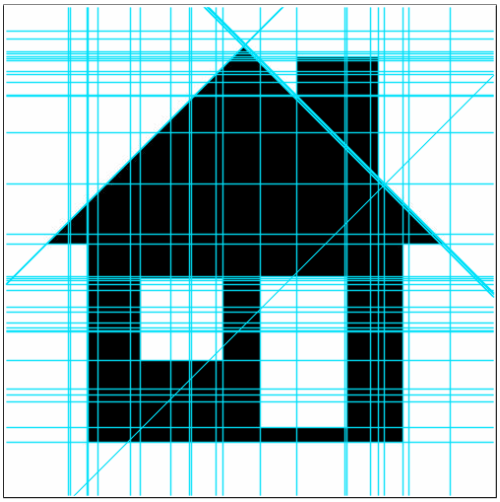
تصویر ورودی $\sigma = 67$



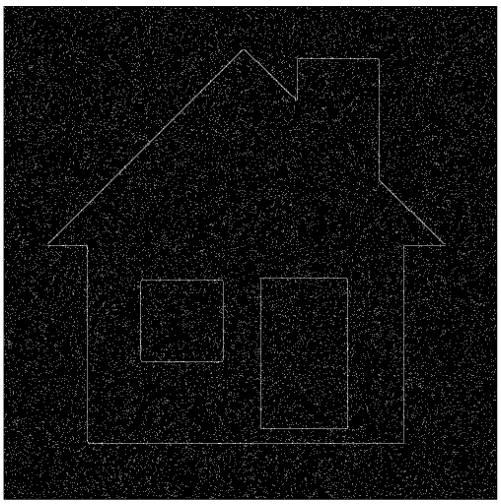
هاف: $\theta = 1^\circ, r = 0.1, threshold = 50$



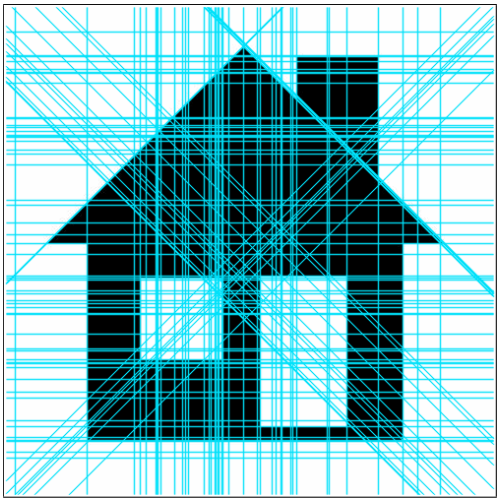
تصویر ورودی $\sigma = 69$



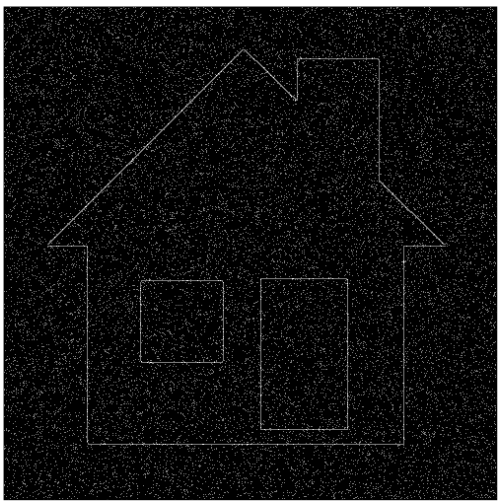
هاف: $\theta = 1^\circ, r = 0.1, threshold = 50$



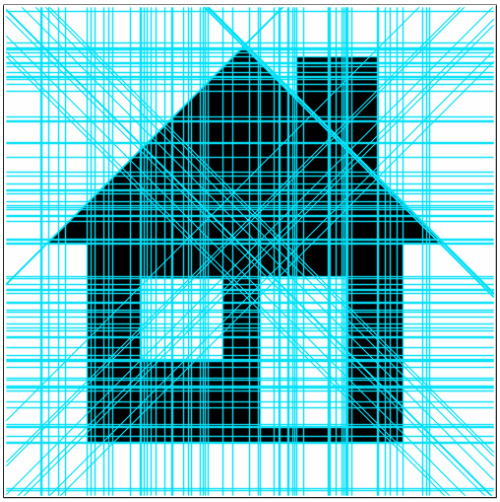
تصویر ورودی $\sigma = 70$



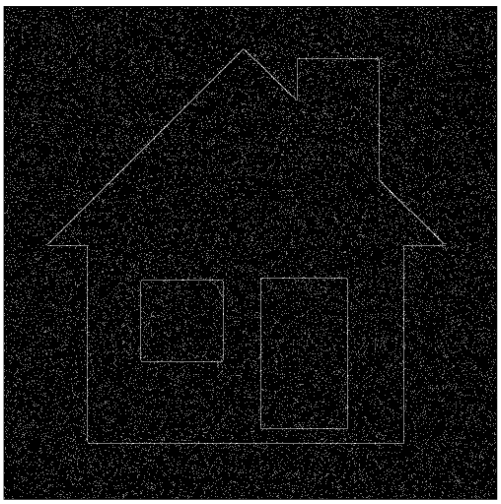
هاف: $\theta = 1^\circ, r = 0.1, threshold = 50$



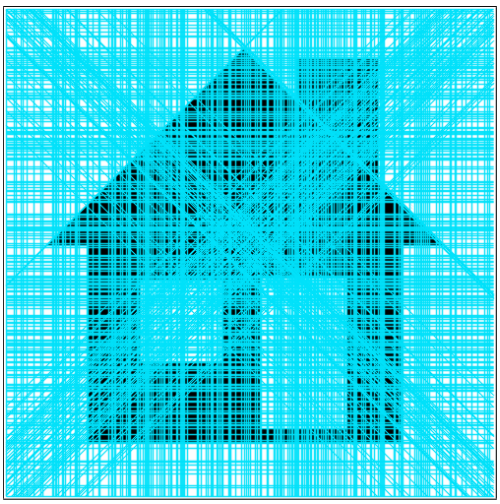
تصویر ورودی $\sigma = 71$



هاف: $\theta = 1^\circ, r = 0.1, threshold = 50$



تصویر ورودی $\sigma = 75$



هاف: $\theta = 1^\circ, r = 0.1, threshold = 50$

۳.۲ هاف دایره

پرسش: تبدیل دایره هاف را بر تصویری شامل اشکال دایره‌ای اعمال کرده و پارامترهای آن را به گونه‌ای تنظیم کنید که دایره‌ها را بیابد.

پاسخ:

با اعمال فیلتر گاسی رفع نویز با مقادیر پارامترهای زیر:

GaussianBlur(img, img, Size(5, 5), 1,1);

و هاف دایره با پارامترهای زیر:

HoughCircles(img, circles, CV_HOUGH_GRADIENT, 0.1,
img.rows/20, 100, 39, 1,100);

دایره‌های موجود در تصویر زیر را پیدا کردیم.

تصویر ورودی

هاف دایره

۳ بخش سوم: کانتورها

۱.۳ کانتور

پرسش: کانتور چیست؟ در OpenCV چه توابعی برای پیدا کردن کانتور و یافتن خصوصیات آن تعریف شده است؟ به طور خلاصه توضیح دهید.

پاسخ:

به طور کلی، کانتور یک منحنی روی یک تابع دو متغیره است که تابع در روی این منحنی، مقادیر یکسانی دارد. حال در تصویر، به هر منحنی که روی آن، پیکسل‌ها یک خاصیت را داشته‌باشند (نظیر رنگ، شدت روشنایی و یا گرادیان)، یک کانتور می‌گویند. از خصوصیات کانتورها که در OpenCV به‌دست می‌آید می‌توان به تعداد آن‌ها، سلسله‌مراتب یا ترتیب آن‌ها، فرزندان و والدین هر کانتور و نیز کانتورهای هم‌سطح آن، اشاره کرد.

۲.۳ findContours

پرسش: تابع findContours از چه روشی برای تعیین پیرامون استفاده می‌کند؟ پارامترهای این تابع چیست؟ این تابع را بر روی یک تصویر اعمال کرده و نتیجه را نشان دهید.

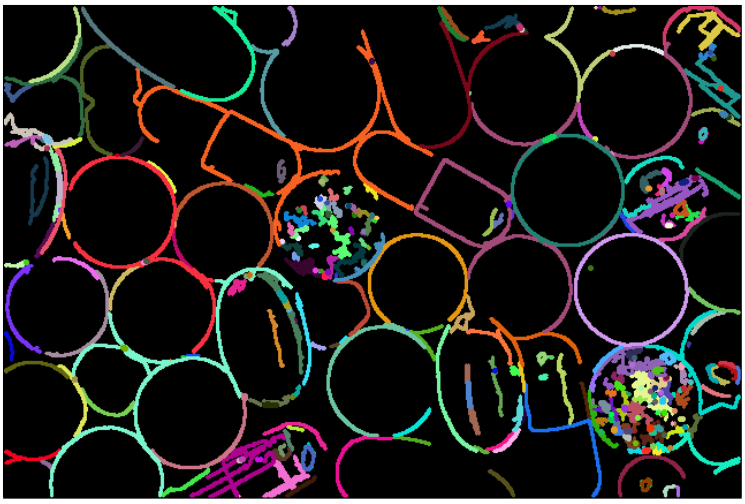
پاسخ: با استفاده از خروجی لبه‌یاب کنی، از روش‌های زیر استفاده می‌کند:

CV_CHAIN_APPROX_NONE : در این روش هر همسایگی پیکسل برای وجود برروی کانتور بررسی می‌شود.

CV_CHAIN_APPROX_SIMPLE : در راستای عمودی، افقی و قطری، پیکسل‌های روی یک قطعه را فشرده می‌کند و تنها نقاط انتهایی را باقی می‌گذارد.

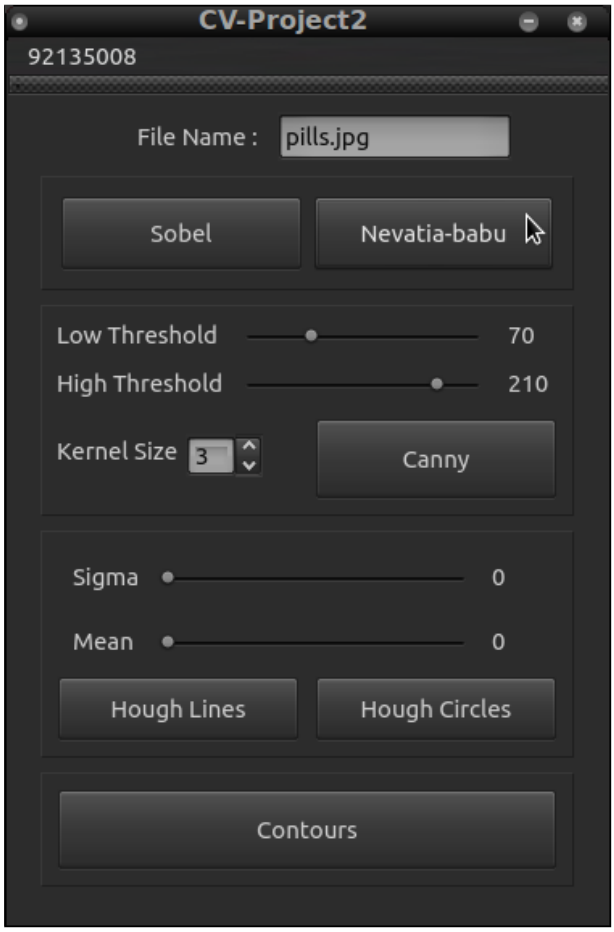
روش مورد استفاده، در پارامترهای ورودی تابع، داده می‌شود. پارامتر مهم دیگر این تابع، mode می‌باشد که تعیین‌کننده‌ی چگونگی محاسبه‌ی کانتورهاست. برای نمونه، فقط کانتورهای بیرونی محاسبه شوند یا همه‌ی کانتورها. نتیجه‌ی اعمال این تابع را در تصویر زیر مشاهده می‌فرمایید که در آن کانتورها با رنگ‌های متفاوت نمایش داده شده‌اند. به آسانی متوجه خواهید شد در دو قرص صورتی‌رنگ و در کپسول تیره‌رنگ پایین صفحه، نواحی مشابه تصویر، در یک کانتور آمده‌اند. دقت شود که این کانتورها، با توجه به خروجی لبه‌یاب کنی محاسبه شده‌اند. این عمل، خود به نوعی تقطیع تصویر مبتنی بر لبه می‌باشد.

تصویر ورودی



کانتورها

تصویری از محیط نرم‌افزار طراحی‌شده: [۱]



[۱] Based on Qt 5.2.1 (GCC 4.8.2 20140206 (prerelease), 64 bit) - C++11