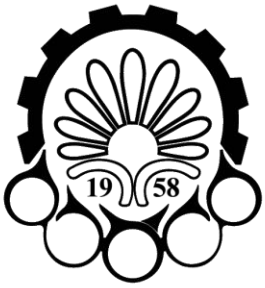


Amirkabir University of Technology
(Tehran Polytechnic)

Department of
Computer Engineering

درس: بینایی ماشین

تمرین دوم

نجمه محمدباقری

۹۹۱۳۱۰۰۹

سوال اول:

تحلیل پارامترها:

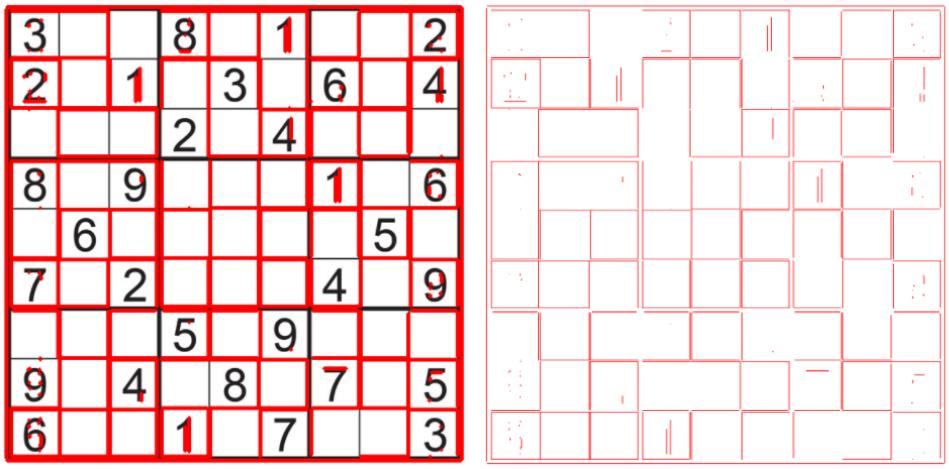
پارامتر theta و rho بیانگر محورهای فضای پارامترها هستند. یعنی با تغییر این دو پارامتر اندازه‌ی واحدها در این دو محور تغییر میکند. همانطور که میدانیم هر نقطه از یک خط در فضای RGB به یک خط در فضای پارامترها نگاشت می‌شود. بنابراین هرچقدر اندازه‌ی این دو پارامتر بزرگتر باشد دقت در فضای پارامترها کمتر است. پارامتر threshold بیانگر تعداد خطوطی است که لازم است در یک نقطه با هم تقاطع داشته باشند تا آن نقطه بعنوان پارامترهای یک خط در فضای rgb در نظر گرفته شود.

در آزمایشات مختلف مشاهده شد که با افزایش پارامترهای theta, rho خطوطی برای اعداد استخراج نمی‌شود اما با کاهش این دو پارامتر خطوط صافی که در ارقام ۱و ۴ وجود دارد مشخص میشود. البته لازم به ذکر است که این دو پارامتر به میزان thereshold نیز وابسته است. هرچقدر این پارامتر کمترباشد در rho, theta ی بالا تعدادی از خطوط اصلی حذف می‌شوند و هرچقدر thereshold کمتر باشد خطوط اضافه تر پیدا می‌شود.

در این تصویر بدلیل اینکه هدف ما تشخیص خطوط افقی و عمودی بود تنظیم پارامتر theta بر مقادیر کوچک تاثیر چندانی نداشت. به همین دلیل این پارامتر را بر روی ۹۰ درجه تنظیم کردیم و بیشتر دو پارامتر دیگر را تغییر دادیم.

در تصویر زیر بهترین حالت را با تنظیم این ۳ پارامتر مشاهده می‌کنیم:

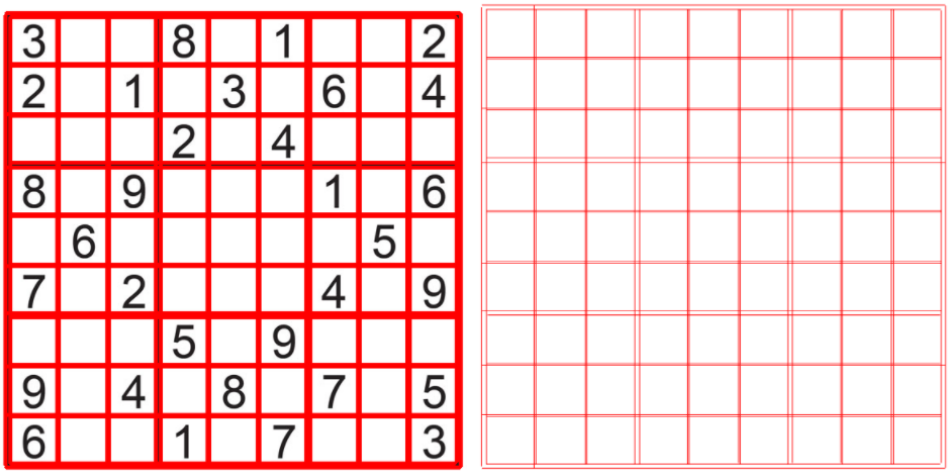
```
Rho = 0.01, theta = 90*np.pi/180, threshold = 50
```



تابع استفاده شده همانطور که در صورت تمرین ذکرشده است houghlineP() میباشد که این تابع یک تبدیل هاف احتمالاتی است. یعنی از تمام نقاط روی لبه برای استخراج خطوط استفاده نمی‌شود. یک زیرمجموعه تصادفی از نقاط روی لبه برای تبدیل هاف استفاده میشود. در نتیجه سرعت الگوریتم بالا می‌رود اما ممکن است خطوط به صورت ناقص شناسایی شوند. به همین دلیل دو پارامتر دیگر به این تابع اضافه‌شده است تا با استفاده از این دو پارامتر بتوان عملکرد تبدیل هاف را در بهترین حالت نگه داشت. این دوپارامتر طول کوتاه‌ترین خط و بیشترین فاصله‌ی بین دو خط را مشخص می‌کنند.

در حالت دوم از دو پارامتر minlinelength, maxlinegap نیز استفاده شده است. در این حالت خروجی بهتر از حالت قبل است. زیرا پارامترهای قبلی را کنترل و خطوط کوچک را حذف می‌کنند. نتایج در شکل زیر قابل مشاهده است.

```
Rho = 1, theta = 90*np.pi/180, threshold =
300, minLineLength =10, maxLineGap = 50
```



سوال دوم:

سرعت عملکرد این تبدیل خوب است و زمان‌بر نیست. اما نکته‌ی کلیدی در استفاده از این تابع، تنظیم دقیق پارامترهاست. با تغییر اندک پارامترها دقت به شدت تغییر می‌کند. پارامترها در این تابع بصورت زیر است:

روش تشخیص: hough_gradiant

Dp: مقیاس تصویر (برای تغییر اندازه‌ی تصویر)

mindst: کمترین فاصله‌ی بین دو مرکز دایره

Param1: پارامتر حد آستانه در تابع canny

Param2: حد آستانه تعداد تقاطع‌ها در فضای پارامترها برای تشخیص دایره‌ها

minRadius: شعاع کوچک‌ترین دایره

maxRadius: شعاع بزرگ‌ترین دایره.

```
Dp = 1, mindst = 500, param1=150, param2=10, minRadius=30, maxRadius=40
```

خروجی این قسمت در پوشه‌ی تمرین ضمیمه شده است.

سوال سوم:

برای پیدا کردن بیضی با روش هاف از کتابخانه‌ی skimage استفاده شده است. قبل از اعمال الگوریتم ابتدا تصویر را سطح خاکستری کردیم و برای حذف نویز و هموارسازی یک gaussian_blur بر روی تصویر سطح خاکستری اعمال می‌کنیم. سپس لبه‌های تصویر را با استفاده از canny بدست میاوریم و این لبه‌ها را به الگوریتم هاف برای تشخیص بیضی میفرستیم.

لازم بذکر است برای دستیابی به دقت و عملکرد بالاتر لازم است در تصویر ورودی قسمت پیراهن، گردن و شانه‌ها نباشد. همچنین موها به گونه‌ای باشد که در زمان تشخیص لبه‌ها، لبه‌های موها حالت بیضی را خراب نکند.

نکته‌ی مهم دیگر در این الگوریتم سرعت بسیار کند آن بود که برای حل این مشکل فریم‌های ورودی را تغییر اندازه دادیم و بسیار کوچک کردیم.

پارامترهای مهم این تابع طول بزرگترین قطر، کوچکترین قطر، حد آستانه و دقت است. هرچقدر دقت بالاتر باشد یعنی اندازه‌ی واحدها در فضای پارامتری کوچکتر است.

خروجی در زمان تحویل نشان داده می‌شود.

سوال چهارم:

الگوریتم هاف تا حد خیلی خوبی نسبت به مقیاس و چرخش مقاوم است. در آزمایشات انجام‌شده در قسمت قبل، مشاهده کردیم که با تغییر زاویه‌ی سر و تغییر مقیاس الگوریتم قادر به پیدا کردن بیضی می‌باشد. اما سرعت الگوریتم به شدت پایین است. همانطور که در قسمت قبل گفته شد برای حل این مشکل تصویر ورودی را تغییر اندازه به ابعاد ۱۰۰*۱۰۰ دادیم. اما با این کار میزان زیادی از اطلاعات از دست می‌رود. بنابراین برای حل این

مشکل لازم است تغییری در الگوریتم هاف ایجاد شود که سرعت آن افزایش یابد تا نیاز به کوچک کردن تصویر ورودی نباشد.

یک راه حل کوچک کردن فضای جستجوی پارامترهاست. بدین منظور می‌توانیم از اطلاعات ساختار چهره استفاده کنیم و چرخش چهره را خنثی و به یک درجه‌ی ثابت برسانیم. در این حالت لازم نیست به ازای تمام جهات تبدیل هاف محاسبه شود و فضای جستجوی به ازای پارامتر orientation کاهش می‌یابد. همچنین می‌توان با استفاده از هرم دقت مقیاس‌ها مختلف را به یک حالت استاندارد تبدیل کرد تا الگوریتم لازم نباشد تمام بیضی‌های با ابعاد مختلف را پیدا کند. در این حالت نیز فضای جستجوی پارامترها کاهش می‌یابد.

در این کاربرد خاص میتوان اطلاعات ساختاری چهره را با استفاده از فاصله‌ی بین دو چشم و روابطی که بین این فاصله و عرض چهره وجود دارد استخراج کرد.

بنابراین راه‌حل کلی، اعمال یک سری پیش‌پردازش بر روی تصویر اولیه (مانند چرخش تصویر و تغییر مقیاس) و محدود کردن پارامترها است.