گزارش تمرین ششم بینایی ماشین

نام تهیه‌کننده: ملیکا نوبختیان

شماره دانشجویی: ۹۷۵۲۲۰۹۴

نسخه: ۱

۱- سوال اول

چون تنها می‌خواهیم معادله وتر را بیابیم، تمام نقاط اضلاع های دیگر هم همچنین نقاط نویزی تشخیص داده شده outlier محسوب خواهند بود و تنها ۱۲۰ نقطه روی وتر inlier هستند.

$$w = \frac{120}{120 + 60 + 80 + 100} = \frac{120}{360} = \frac{1}{3}$$

حالت اول: $p = 0.9$

$$k = \frac{\log(1-p)}{\log(1-w^2)} = \frac{\log 0.1}{\log 0.9} \approx 21.85 \rightarrow 21 \Rightarrow \boxed{k = 21}$$

اگر بخواهیم با احتمال ۹۰٪ درصد وتر را به دست آوریم، لازم است تا RANSAC را حداقل ۲۱ بار اجرا کنیم.

حالت دوم: $p = 0.99$

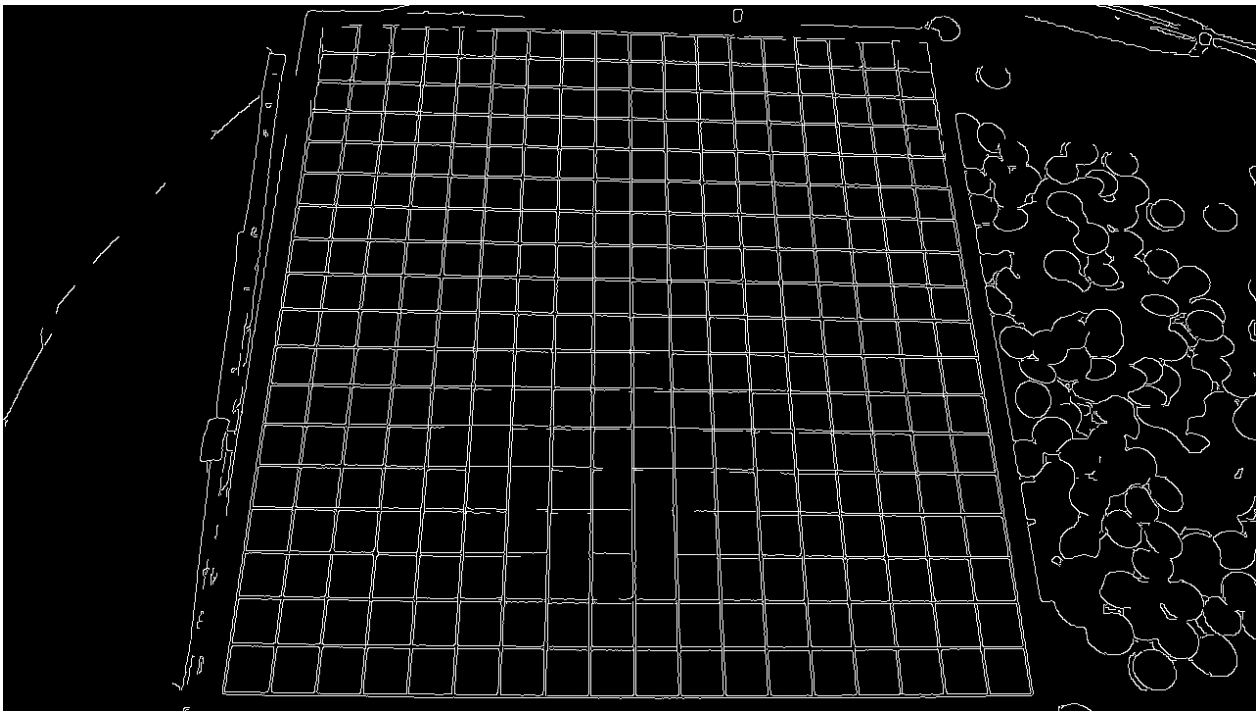
$$k = \frac{\log 0.01}{\log 0.9} \approx 43.7 = 43 \rightarrow \boxed{k = 43}$$

اگر بخواهیم با احتمال ۹۹٪ وتر را به دست آوریم، لازم است تا RANSAC را حداقل 43 بار اجرا کنیم.

۲- سوال دوم

در ابتدا لازم است با استفاده از canny نقاط لبه را شناسایی کنیم و سپس روی آن‌ها hough را اجرا کنیم:

```
img = cv2.imread('/content/LineDetection.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
edges = cv2.Canny(gray, 150, 300)
```

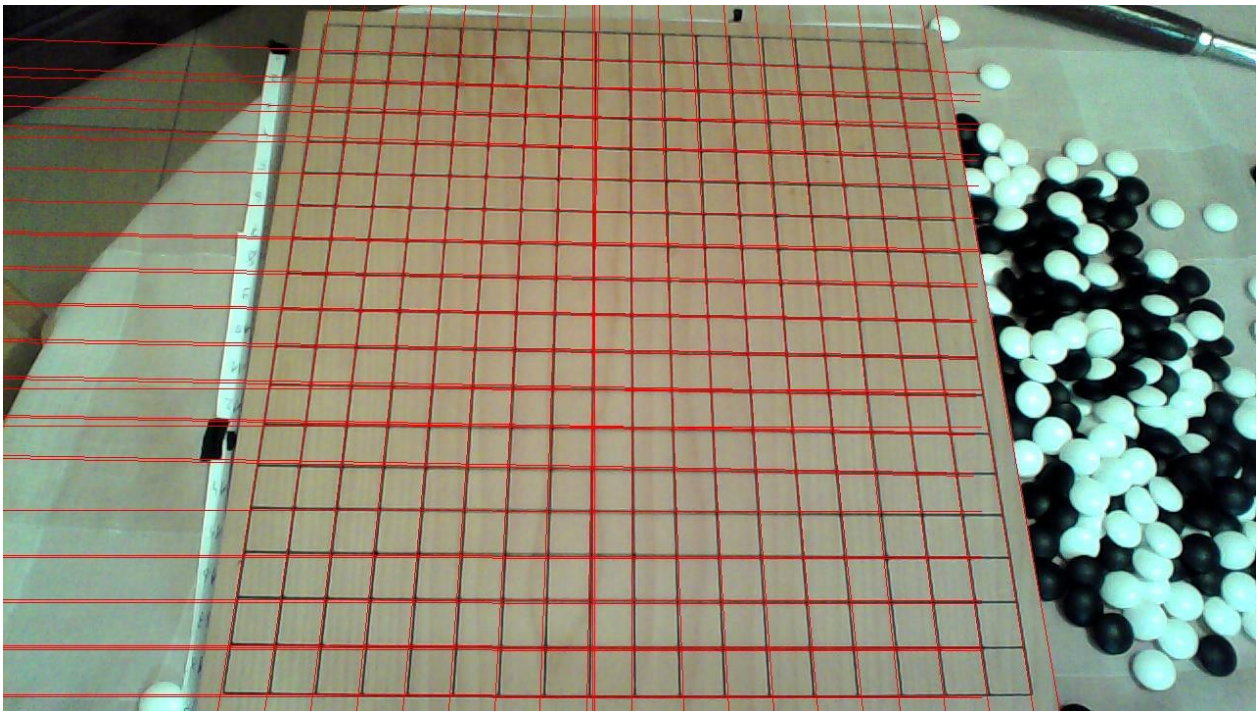


سپس تبدیل hough را روی این تصویر اعمال می‌کنیم و خط‌های به دست‌آمده را روی تصویر اولیه رسم می‌کنیم:

```
lines = cv2.HoughLines(edges, 1, np.pi / 180, 250)
# Draw detected lines
for i in range(0, len(lines)):
    rho = lines[i][0][0]
    theta = lines[i][0][1]
    a = math.cos(theta)
    b = math.sin(theta)
    x0 = a * rho
    y0 = b * rho
    pt1 = (int(x0 + 1000*(-b)), int(y0 + 1000*(a)))
    pt2 = (int(x0 - 1000*(-b)), int(y0 - 1000*(a)))
    cv2.line(img1, pt1, pt2, (0,0,255), 1)
```

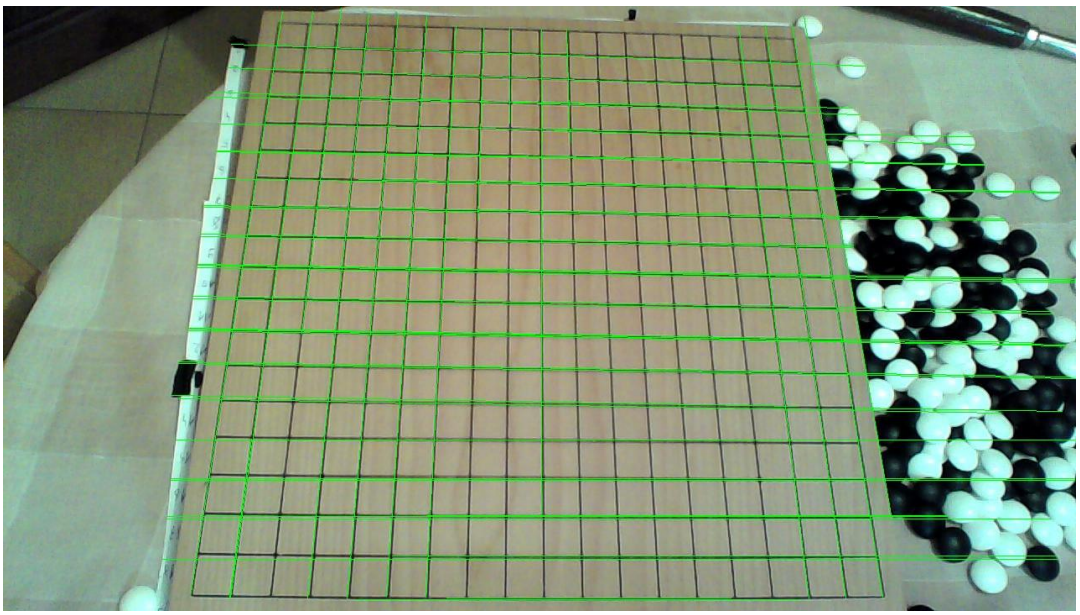
ورودی اول تابع Houghlines تصویر gray scale ای است که لبه‌های آن تشخیص داده شده‌است. ورودی دوم ما مقیاس پارامتر ρ یا همان rho را نشان می‌دهد که ما مقیاس ۱ پیکسل را برای آن در نظر گرفتیم. پارامتر بعدی مقیاس زاویه یا θ را مشخص می‌کند که برای این قسمت نیز مقدار یک درجه را در نظر گرفتیم. پارامتر بعد نیز همان threshold برای رای گیری را مشخص می‌کند که این مقدار را برابر ۲۵۰ گذاشتیم.

پس از تشخیص خطوط با استفاده از الگوریتم hough این خطوط را روی تصویر رسم کرده‌ایم که نتیجه آن به صورت زیر خواهد بود:



اگر از probabilistic hough transform استفاده کنیم و خطوط را بکشیم نتیجه به شکل زیر خواهد بود:

```
# Detect points that form a line
lines = cv2.HoughLinesP(edges, 1, np.pi / 180, 250, minLineLength=5, maxLi
neGap=80)
# Draw lines on the image
for line in lines:
    x1, y1, x2, y2 = line[0]
    cv2.line(img2, (x1, y1), (x2, y2), (0, 255, 0), 1)
```



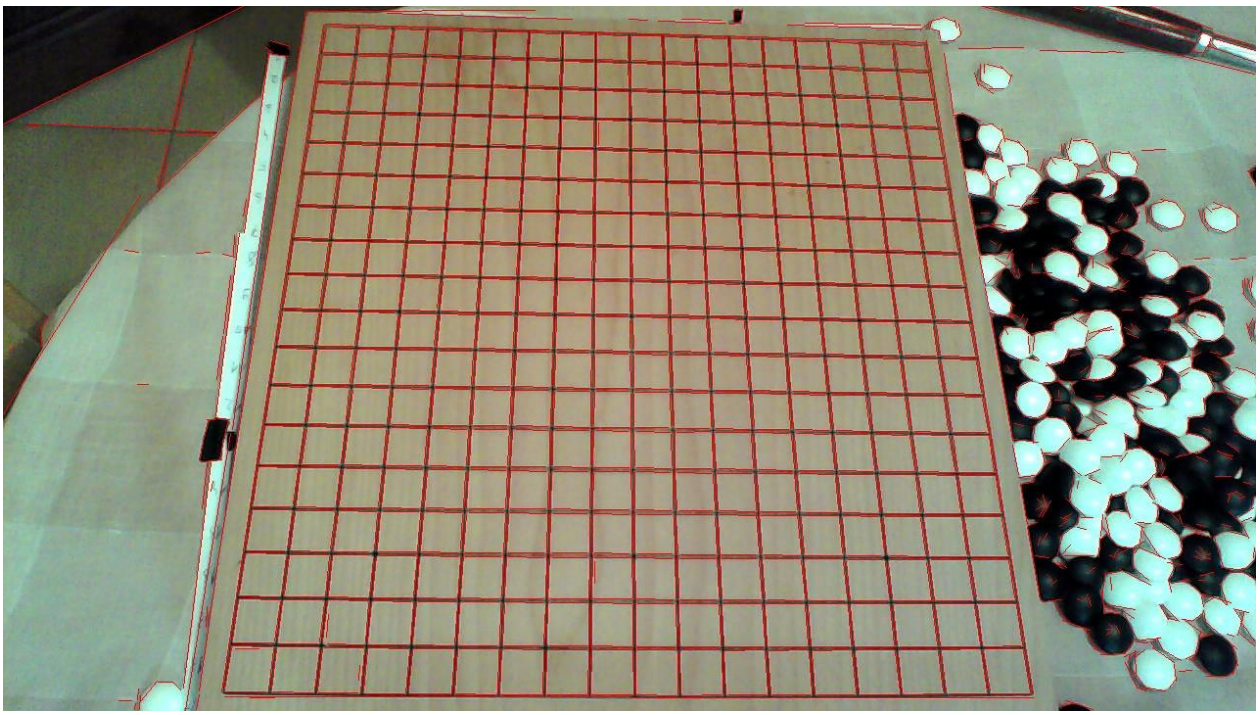
در این متد نسبت به متد قبلی دو ورودی جدید خواهیم داشت. minLineLength مشخص می‌کند که حداقل با چه تعداد از نقاط می‌توانیم یک خط را تشکیل دهیم. maxLineGap نیز مشخص می‌کند نقاطی که روی یک خط هستند حداکثر می‌توانند چقدر از یکدیگر فاصله داشته باشند.
من کمترین برای کمترین تعداد نقاطی که یک خط را تشکیل می‌دهند مقدار ۵ و برای حداکثر فاصله مقدار ۸۰ را درنظر گرفتم. این متد با سعی بر پیداکردن ابتدا و انتها خطوط، می‌خواهد حدود اصلی خط را مشخص کند.

منبع:

https://docs.opencv.org/3.4/d9/db0/tutorial_hough_lines.html

۳- سوال سوم

در این قسمت از الگوریتم LSD یا LineSegmentDetector استفاده شده‌است. این الگوریتم برای تشخیص پاره‌خط به کار می‌رود و برای این کار به خوبی از جهت گرادیان استفاده می‌کند. به این صورت که نقاطی که هم به هم نزدیک هستند و هم زاویه گرادیان آن‌ها نزدیک به هم است به عنوان پاره‌خط شناخته خواهند شد. نتیجه اعمال این روش روی تصویر به شکل زیر است:



نتیجه نسبت به تبدیل hough قابل لمس زیرا حدود خطوط و تشخیص درست خطوط واقعی بهتر از hough انجام شده‌است و این بهبود به دلیل استفاده درست از جهت گرادیان است. هم چنین برخلاف حالت قبل که خطوط نزدیک به هم و با فاصله کم زیاد بودند در اینجا کمتر این اتفاق را مشاهده می‌کنیم.

۴- سوال چهارم

در ابتدا برای تبدیل rgb به cmyk باید مقادیر r, g و b را scale کنیم:

$$R' = R/255$$

$$G' = G/255$$

$$B' = \mathrm{B}/255$$

پس از آن ابتدا مقدار k را با استفاده از فرمول زیر به دست می‌آوریم:

$$K = 1\text{-}\max(R', G', B')$$

سپس با داشتن مقدار k و مقادیر scale شده C, M و Y را به دست می‌آوریم:

$$C = (1\text{-}R'\text{-}K) \,/\, (1\text{-}K)$$

$$M = (1\text{-}G'\text{-}K) \,/\, (1\text{-}K)$$

$$Y = (1\text{-}B'\text{-}K) \,/\, (1\text{-}K)$$

در آخر لازم است این مقادیر را در scale مورد استفاده cmyk یعنی ۱۰۰ ضرب کنیم. کد آن نیز به صورت زیر خواهد بود:

```
def rgb_to_cmyk(r, g, b, RGB_SCALE = 255, CMYK_SCALE = 100):

  #TODO
  scaled_r = r / RGB_SCALE
  scaled_g = g / RGB_SCALE
  scaled_b = b / RGB_SCALE

  k = 1 - max(scaled_r, scaled_g, scaled_b)
  c = 1 - k - scaled_r
  m = 1 - k - scaled_g
  y = 1 - k - scaled_b

  c = c / ( 1 - k )
  m = m / ( 1 - k )
  y = y / ( 1 - k )

  return int(c * CMYK_SCALE), int(m * CMYK_SCALE), int(y * CMYK_SCALE), in
t(k * CMYK_SCALE)
```

برای برگرداندن cmyk به rgb نیز مانند قدم اول تبدیل قبل لازم است مقادیر cmyk را scale کنیم که برای این کار لازم است مقادیر آن‌ها را به ۱۰۰ تقسیم کنیم. سپس با استفاده از فرمول‌های زیر تبدیل را انجام می‌دهیم:

$$R = 255 \times (1-C) \times (1-K)$$

$$G = 255 \times (1-M) \times (1-K)$$

$$B = 255 \times (1-Y) \times (1-K)$$

کد آن نیز به صورت زیر خواهد بود:

```
def cmyk_to_rgb(c, m, y, k, CMYK_SCALE = 100, RGB_SCALE = 255):

  #TODO
  r = int(RGB_SCALE * (1 - c / CMYK_SCALE) * (1 - k / CMYK_SCALE))
  g = int(RGB_SCALE * (1 - m / CMYK_SCALE) * (1 - k / CMYK_SCALE))
  b = int(RGB_SCALE * (1 - y / CMYK_SCALE) * (1 - k / CMYK_SCALE))

  return r, g, b
```

منابع:

https://www.rapidtables.com/convert/color/cmyk-to-rgb.html
https://www.rapidtables.com/convert/color/rgb-to-cmyk.html

۵- سوال پنجم

با استفاده از فرمول‌های موجود در اسلایدها به تبدیل مقادیر rgb پرداختم. اما قبل از این تبدیل‌ها باید scaleکردن مقادیر r,g و b فراموش نشود. هم چنین مقدار theta به دست آمده قبل از اینکه به H تبدیل شود در ضریب 255/360 ضرب خواهد شد. کد آن به شکل زیر است:

```
def transform_rgb(r, g, b, RGB_SCALE = 255):

    r = r / RGB_SCALE
    g = g / RGB_SCALE
    b = b / RGB_SCALE

    I = (r + g + b) / 3

    theta = math.acos(((r - g) + (r - b)) / (2 * (math.sqrt((math.pow(r -
g, 2) + (r - b)*(g - b))))))
    theta = theta * 255 / 360
    H = theta if b <= g else 360 - theta

    S = 1 - ((3 * min(r, g, b)) / (r + g + b))
```

```
    V = max(r, g, b)
    L = (max(r, g, b) + min(r, g, b)) / 2

    return I * 100, H, S * 100, V * 100, L * 100
```

نتیجه این تابع برای مقادیر rgb گفته شده در سوال به شکل زیر است. مقادیر I, S, V, L به شکل درصد بیان شده‌اند و مقدار H هم به شکل زاویه است:

```
I :  54.248366013071895
H :  358.99259698317417
S :  53.01204819277109
V :  78.43137254901961
L :  51.96078431372548
```