۱. شدت روشنایی یک پیکسل ($f(x,y)$) با دو جزء مشخص می‌شود: $f(x,y) = i(x,y)r(x,y)$

۱. $i(x,y)$ میزان روشنایی که از منبع نور به صحنه تابیده شده است.

۲. $r(x,y)$ میزان روشنایی که اشیاء موجود در صحنه منعکس می‌کنند.

همچنین در هنگام عکس‌برداری **با دوربین** عوامل زیر بر میزان روشنایی تصویر ثبت‌شده تاثیر دارند:

۱. **سرعت شاتر**: هرچه دریچه دوربین کندتر باز و بسته شود، حسگر دوربین انرژی نوری بیشتری دریافت می‌کند و تصویر ثبت‌شده روشن‌تر خواهد بود.

۲. **اندازه دریچه دوربین**: هرچه دریچه دوربین بزرگ‌تر باشد نور بیشتری وارد می‌شود و تصویر ثبت‌شده روشنایی بیشتری خواهد داشت.

۲. عنوان ایده: انتخاب محصول دلخواه هنگام خرید از دستگاه vending machine فقط با نگاه به تصویر محصول

شرح ایده: برخی از دستگاه‌های فروش خودکار (vending machine) به‌دلیل تنوع محصولاتی که عرضه می‌کنند سهولت استفاده ندارند و برخی استفاده‌کنندگان (مخصوصا سالمندان) را بخاطر وجود تعداد زیاد دکمه‌هایی که دارند دچار سردرگمی می‌کنند. در ایده‌ی من خریدار در محل تعیین شده می‌ایستد و صرفا با نگاه کردن به محصول موردنظرش (یا تصویر محصول موردنظرش که می‌تواند روی دیوار پشت دستگاه یا کنار دستگاه قرار بگیرد) آن را انتخاب می‌کند. سپس دستگاه از او می‌پرسد آیا محصول تشخیص‌داده‌شده توسط دستگاه صحیح است یا نه و پس از فشردن دکمه تایید می‌تواند پرداخت کند. در تمام فرآیند خرید، خریدار فقط یک دکمه را برای انتخاب محصول خود فشار می‌دهد و آن هم برای تایید محصول موردنظرش و این سهولت کار با دستگاه فروش را بسیار برای او بالا می‌برد. این ایده علاوه بر جالب و جذاب بودن برای خریدارانی که اولین بار با آن مواجه می‌شوند، می‌تواند به سالمندان یا کودکان در استفاده ساده‌تر از دستگاه کمک کند، بدلیل وجود دکمه‌های کمتر استهلاک سخت‌افزاری کمتری دارد و به لحاظ بهداشتی، مشکل انتقال بیماری و آلودگی‌ها از طریق دکمه‌های دستگاه را به حداقل می‌رساند و زحمت و هزینه نظافت آن را هم کم‌تر می‌کند.

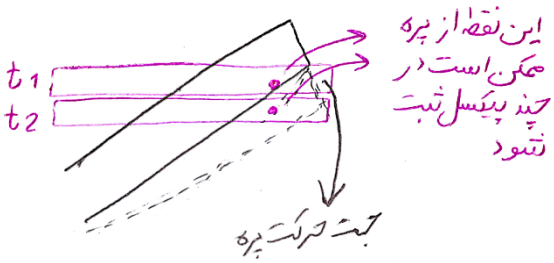
توضیح استفاده از بینایی ماشین در ایده گفته شده: برای تشخیص محصول انتخاب‌شده کاربر با فرض اینکه کاربر در مکان مشخص‌شده ایستاده است با پردازش تصویر ورودی از سنسور دستگاه و تشخیص جهت نگاه او، با توجه به مکان قرار داده شده محصول (یا تصویر آن) می‌توانیم تشخیص دهیم به کدام محصول (یا تصویر آن) نگاه می‌کند.

۳.الف. در حسگر خطی نوع عملکرد به این شکل است که تعدادی حسگر بصورت ردیفی در کنار هم قرار گرفته‌اند و قرار است با حرکت فقط افقی یا فقط عمودی کل صحنه را ثبت کنند. مثلا برای ثبت تصویر ۱۰۰۰در۱۰۰۰ اگر ۱۰۰۰حسگر بصورت افقی در کنار هم حسگر خطی ما را تشکیل بدهند آنگاه حسگر خطی باید بصورت عمودی حرکت کند و در هربار یک ردیف افقی از تصویر را ثبت کند. طبیعتا این کار را ۱۰۰۰بار باید انجام دهد تا کل تصویر ثبت شود.

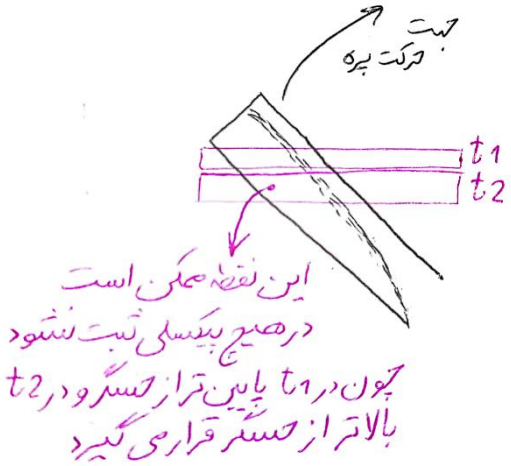
در حسگر آرایه‌ای می‌توانیم ۱ میلیون حسگر جاگذاری کنیم تا همگی همزمان تصویر صحنه را ثبت کنند. طبیعتا استفاده از حسگر آرایه‌ای بسیار سریعتر از حسگرهای خطی است چون نیاز به جابجایی حسگر نیست اما بسیار گران‌تر است چون تعداد حسگرها نسبت به حسگر خطی از ۱۰۰۰تا به ۱میلیون‌تا افزایش پیدا کرده است.

۳.ب. در حسگر آرایه‌ای تصویر مانند همان چیزی که در pic.jpg داده شده ثبت خواهد شد چون به صورت همزمان حسگرها پیکسل‌های مختلف را ثبت می‌کنند و تاثیر حرکت پره‌های آسیاب در تصویر ثبت‌شده دیده نخواهد شد.

اما وقتی از حسگر خطی استفاده کنیم تصویر ثبت‌شده بصورتی خواهد بود که پره‌های آسیاب بادی که در حال حرکت هستند موج بر خواهند داشت و بصورت صاف در تصویر ثبت نمی‌شوند. با فرض اینکه حسگر خطی از بالا به پایین حرکت می‌کند می‌توان گفت در سمت راست تصویر که پره‌های آسیاب به پایین در حال حرکت هستند، پره‌ها ضخیم‌تر از چیزی که هستند ثبت می‌شوند و موج رو به پایین برخواهند داشت چون همزمان با حرکت حسگر، پره هم به سمت پایین حرکت می‌کند و ممکن است نقطه‌ای از پره هم در یک پیکسل و هم در پیکسل پایینی‌اش ثبت شود.



بطور مشابه در سمت چپ تصویر پره‌ها نازک‌تر از چیزی که در واقعیت هستند ثبت می‌شوند و بصورت تورفته رو به بالا موج خواهند برداشت.



۱.۴. نصب موفق OpenCV

```
In [2]: !pip install opencv-python
        !pip install opencv-contrib-python

Requirement already satisfied: opencv-python in c:\users\amiri\anaconda3\envs\fcv_mohammadi\lib\site-packages (4.6.0.66)
Requirement already satisfied: numpy>=1.14.5 in c:\users\amiri\anaconda3\envs\fcv_mohammadi\lib\site-packages (from opencv-pyth
on) (1.21.6)
Requirement already satisfied: opencv-contrib-python in c:\users\amiri\anaconda3\envs\fcv_mohammadi\lib\site-packages (4.6.0.6
6)
Requirement already satisfied: numpy>=1.14.5 in c:\users\amiri\anaconda3\envs\fcv_mohammadi\lib\site-packages (from opencv-cont
rib-python) (1.21.6)
```

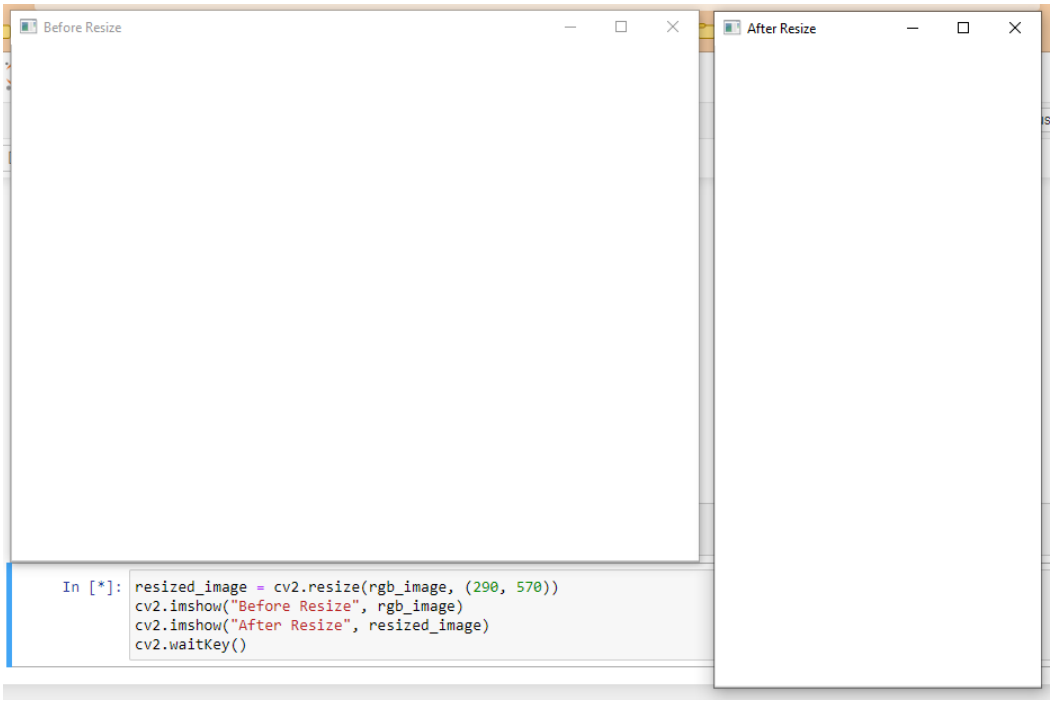
۲.۴. پارامتر اول تابع مسیر فایل تصویر است. در اینجا بدلیل اینکه فایل تصویر در کنار برنامه قرار داده شده است، مسیر فایل بصورت نسبی داده شده است(از root سیستم نوشته نشده) پارامتر دوم نشان‌دهنده نحوه خواندن تصویر است که یکی از سه مقدار cv2.IMREAD_COLOR یا 1 (سه‌کاناله)، cv2.IMREAD_GRAYSCALE یا 0 (تک‌کاناله)، cv2.IMREAD_UNCHANGED یا -1 (چهار کاناله با کانال اضافی alpha برای مشخص کردن transparency) را به خود می‌گیرد. این پارامتر اختیاری است و در صورتی که داده نشود تصویر بصورت پیش‌فرض سه‌کاناله (cv2.IMREAD_COLOR) باز می‌شود. خروجی تابع همانطور که در تصویر روبرو مشخص است پیکسل‌های تصویر هستند که برای مثال در تصویر روبرو ۴۵۹در۶۱۱ پیکسل مختلف هستند که برای هر پیکسل سه کانال B و G و R ذخیره شده‌اند.

```
In [8]: import cv2
        image = cv2.imread("background.png", 1)
        print(image.shape)
        image

(459, 611, 3)

Out[8]: array([[[198, 198, 198],
                [255, 255, 255],
                [255, 255, 255],
                ...,
                [255, 255, 255],
                [255, 255, 255],
                [255, 255, 255]],

               [[198, 198, 198],
                [255, 255, 255],
                [255, 255, 255],
                ...,
```

۳.۴. تصویر بازشده که بصورت پیش‌فرض بصورت BGR ذخیره شده با کد تبدیل COLOR_BGR2RGB تبدیل به حالت ذخیره‌سازی RGB کردم.

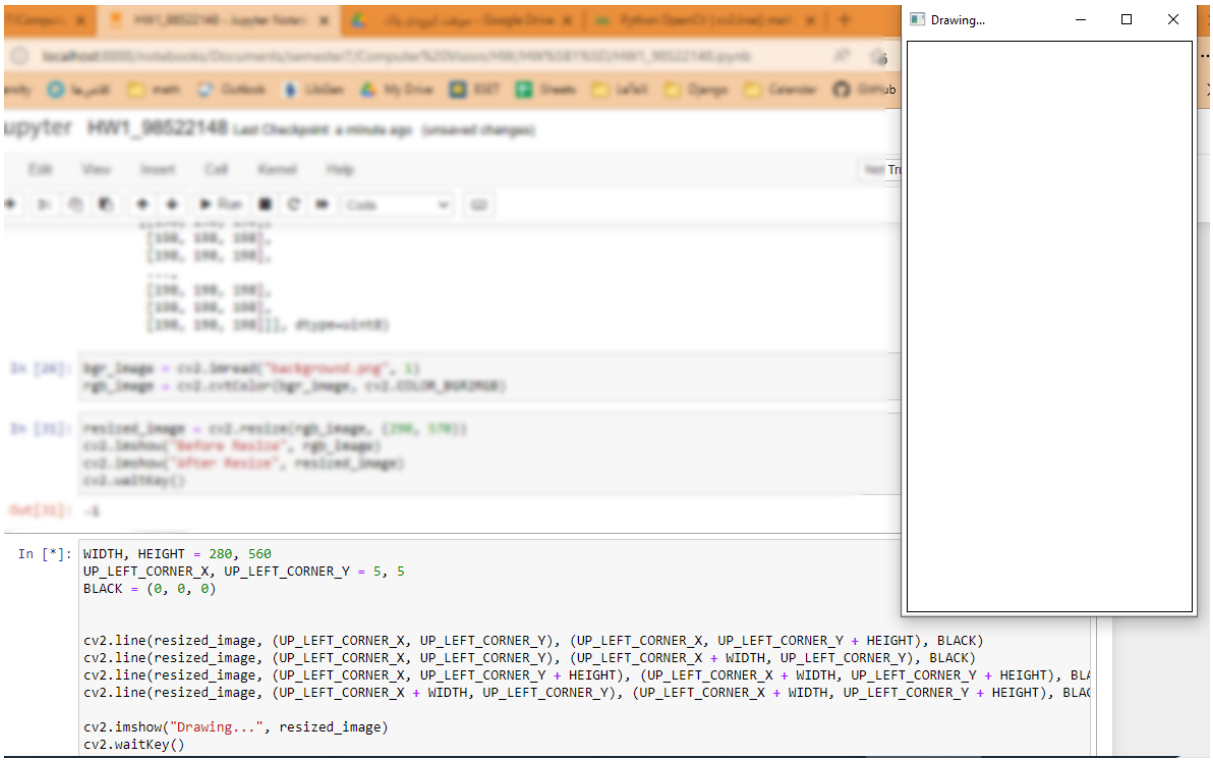
```
In [26]: bgr_image = cv2.imread("background.png", 1)
         rgb_image = cv2.cvtColor(bgr_image, cv2.COLOR_BGR2RGB)
```

۴.۴.



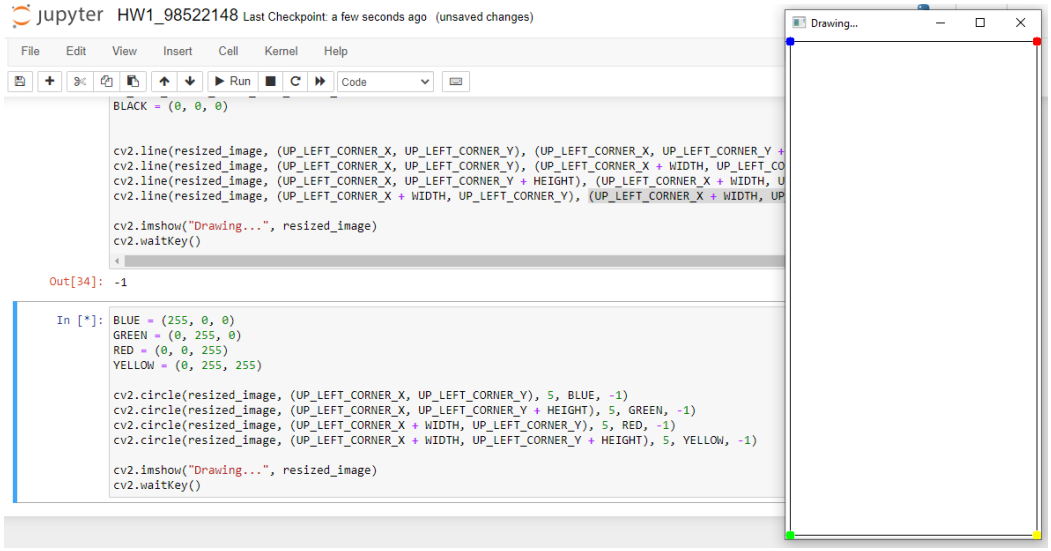
مطابق خواسته سوال با استفاده از تابع resize ابعاد تصویر را به ۲۹۰در۵۷۰ تغییر دادم و سپس با imshow تصویر را قبل از تغییر سایز و بعد از تغییر سایز نمایش دادم.

۵.۴.



مختصات ۴ ضلع مستطیل را (5,5) و (285,5) و (5,565) و (285,565) در نظر گرفتم و در صفحه رسم کردم. سپس تصویر را پس از رسم مستطیل با imshow نمایش دادم.

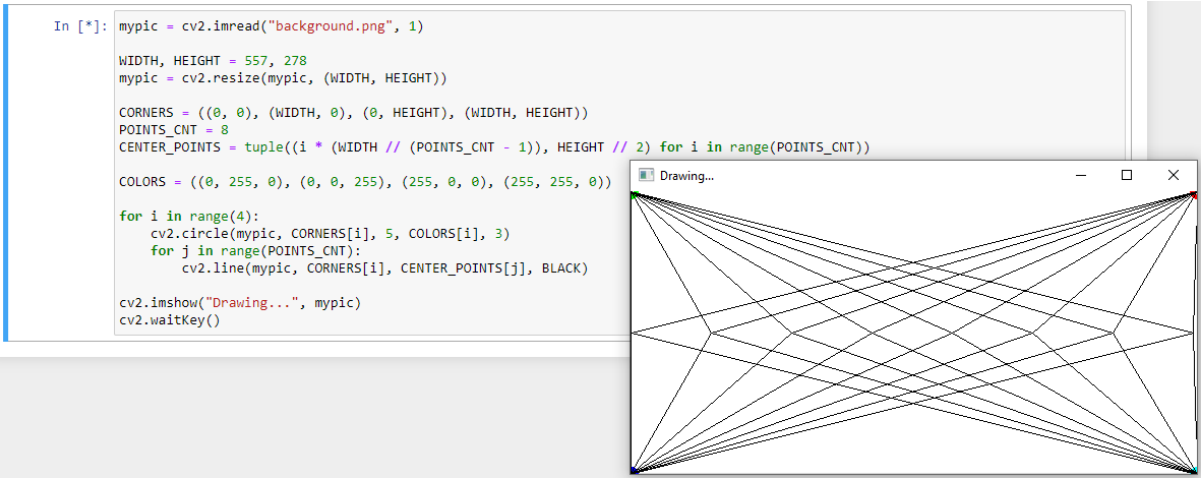
۶.۴



۴ رنگ مختلف را با فرمت BGR ذخیره کردم و سپس از مختصاتی که در قسمت قبل برای کشیدن مستطیل استفاده کرده بودم به عنوان مرکز دایره استفاده کردم. همچنین آرگومان مربوط به ضخامت دایره را -1 دادم تا دایره بصورت توپر رسم شود.

در نهایت هم تصویر نهایی را با imshow نمایش دادم.

۷.۴



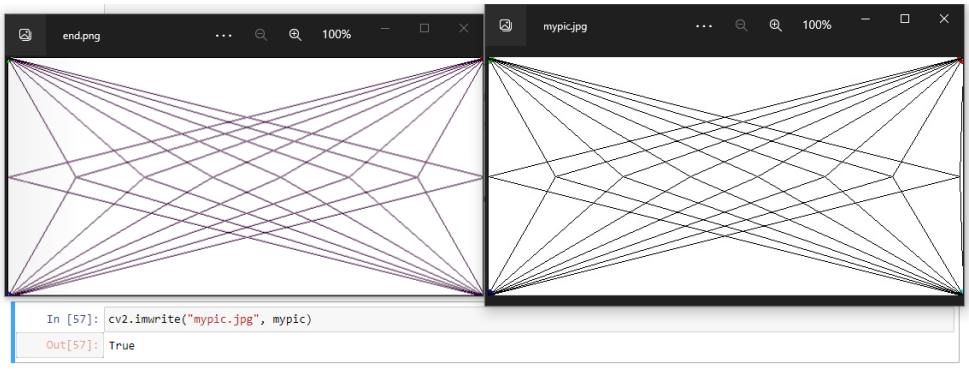
ابتدا همان تصویر background.png را باز کردم و سپس به ابعاد تصویر end.png تغییر ابعاد دادم.

در مرحله بعدی ثوابت موردنیاز را در متغیرها ذخیره کردم:

- CORNERS: مختصات ۴گوشه تصویر
- POINTS_CNT: تعداد نقاطی که در ردیف وسط تصویر، به آن‌ها می‌خواهیم خط بکشیم.
- CENTER_POINTS: مختصات نقاط ردیف وسطی تصویر که از آن‌ها به ۴گوشه تصویر پاره‌خط کشیده شده.
- COLORS: به ترتیب کد رنگ‌های سبز، قرمز، آبی و فیروزه‌ای

در نهایت به ازای هر گوشه ابتدا یک دایره بسیار کوچک رسم کردم (درست مانند end.png) سپس در حلقه‌ی درونی که ۸بار اجرا می‌شود از گوشه‌ی موردنظر به هر یک از ۸ نقطه‌ی ردیف وسطی تصویر یک پاره‌خط رسم کردم.

۸.۴



در نهایت تصویر رسم‌شده را مطابق دستورالعمل تمرین در فایل mypic.jpg ذخیره کردم.

منابع

سوال ۲.۴: Python OpenCV | cv2.imread() method - GeeksforGeeks

سوال ۳.۴: OpenCV: Color Space Conversions

python - Error in OpenCV color conversion from BGR to grayscale - Stack Overflow

سوال ۴.۴: OpenCV Resize image using cv2.resize() (tutorialkart.com)

سوال ۵.۴: Python OpenCV | cv2.line() method - GeeksforGeeks

سوال ۶.۴: Python OpenCV | cv2.circle() method - GeeksforGeeks

سوال ۸.۴: Python OpenCV | cv2.imwrite() method - GeeksforGeeks