Computer vision

Homework 1

Parsa eissazadeh

سوال 1

عوامل تاثیر گذار روی روشنایی یک پیکسل از عکس :

1. منبع نور محیط
2. انعکاس نور از سطحی که عکسش گرفته شده در آن پیکسل .
3. سرعت دریچه : هر چه سرعت بیشتر باشد ، دریچه مدت زمان کمتری باز است و نور کمتری از محیط ضبط می شود .

سوال 2

در بازیافت زباله ، به کمک الگوریتم های بینایی ماشین ، از روی ظاهر زباله ها ، نوع آن را تشخیص می دهیم و در دسته بندی های درست آن ها را بازیافت میکنیم . برای مثال اگر عکسی از قوطی مایع ظرفشویی ای دیدیم که اندکی مچاله شده بود ، به کمک این الگوریتم تشخیص دهیم که برای زباله های پلاستیکی است و آن را در دسته زباله های پلاستیکی بازیافت کنیم .

تقویت دوربین های مدار بر بسته . با استفاده از تصاویری که در طول روز و در روشنایی از محیط دریافت میکنیم ، در شب هم اجسام و محیط را بهتر تشخیص دهیم .

سوال 3

الف)

حسگر آرایه ای یا global shutter از همه ی پیکسل ها یکجا عکس برداری میکند برای همین یک عکس 1000x1000 را در یک واحد زمانی اسکن میکند . ولی با استفاده از یک حسگر خطی ، در یک واحد زمانی 1000 پیکسل که در یک خط هستند اسکن میشوند و این عملیات باید 1000 بار تکرار شود تا کل عکس اسکن شود . در نتیجه 1000 واحد زمانی طول میکشد .

در نتیجه سرعت حسگر آرایه ای هزار برابر سرعت حسگر خطی است و همچنین تعداد حرکات در حسگر خطی هزار برابر بیشتر است .

ب)

با حسگر آرایه ای همه ی پره ها به شکل هم اندازه و هم شکل (در جهت های متفاوت) ضبط می شوند فقط بسته به سرعت شاتر وضوح های متفاوتی می توانند داشته باشند .

اما در حسگر آرایه ای ، اگر از بالا به پایین سنسور ها حرکت کنند . یک پره A در بالا است و به پایین می رود و پره دیگری B در پایین است و به بالا می رود . حالت این دو پره را بررسی میکنیم .

هنگامی که پره A بالا است ، حسگر هم بالا و با هم از بالا به پایین حرکت می کنند . در نتیجه زمان طلاقی مکانیشان بیشتر است و تصویر پره بیشتر ثبت می شود و با طول بیشتری در تصویر نمایان می شود . (منظور از طول این است که سهم بیشتری از دایره آسیاب را اشغال می کند .) بسته به سرعت آسیاب طول نمایان شده از این پره می تواند کمتر و یا بیشتر باشد .

اگر :

- سرعت پره برابر با سرعت حسگر ها باشد (یعنی در مدت زمانی که حسگر ها از بالا به پایین رسیده اند پره هم نصف دایره را زده) ، آن پره به اندازه نصف دایره کشیده خواهد شد .
- اگر سرعت حسگر ها کمتر باشد ، پره به پایین رفته و بالا نیز میرود در نتیجه بیشتر از نصف دایره را اشغال میکند .
- اگر سرعت حسگر ها بیشتر باشد ، پره کمتر از نصف را اشغال می کند زیرا زمانی که حسگر ها به پایین رسیده از بقیه پره ها هم عکس برداری کرده است .

زمانی که پره B پایین است و میخواهد به بالا برود ، حسگر به پایین حرکت میکند و چون تصویری از پره وجود ندارد ، تصویری ضبط نمی کند و پایین می رود . زمانی که حسگر ها و پره ها به هم میرسند و طلاقی دارند ، تصویر پره B ضبط می شود ولی مدتی کوتاه است . در نتیجه زمان ضبط شدن عکس پره کمتر خواهد بود و با طول کمتری در عکس نمایان می شود .

نتیجه نهایی همانند این عکس خواهد شد :

توضیحات بدون منبع هستند ، منبع عکس : [What are Global Shutter and Rolling shutter Cameras? How to choose the one that fits the application?]

سوال 4

توضیح پارامتر های imread :

برگرفته از وبسایت [OpenCV: Flags used for image file reading and writing]

متد imread دو ورودی میگیرد که اولی آدرس عکس است و دومی flag متد . که چند تا متغیر constant هستند که مود متد را تعیین می کنند . متد imread در کل ۱۳ flag دارد . برای مثال :

- ۳ تا از آنها IMREAD_REDUCED_GRAYSCALE در کنار یک عدد توان ۲ از ۲ تا ۸ هستند . که عکس را سیاه و سفید میکنند و سایز را به نسبت همان عددی که دارد کوچک می کند .
- ۳ تا از آنها IMREAD_REDUCED در کنار یک عدد توان ۲ از ۲ تا ۸ هستند . که فقط سایز را به نسبت همان عددی که دارد کوچک می کند .
- IMREAD_COLOR که عکس را به 3 کانال BGR تبدیل میکند . که از همین استفاده میکنیم برای سوال)
-

برای تبدیل کردن به RGB باید از متد cvtColor به گونه زیر استفاده کنیم :

چون ما عکس را در فرمت BGR ذخیره کردیم ازثابت COLOR_BGR2RGB استفاده می کنیم .

```
## convert to RGB
image_rgb = cv2.cvtColor(img , cv2.COLOR_BGR2RGB)
```

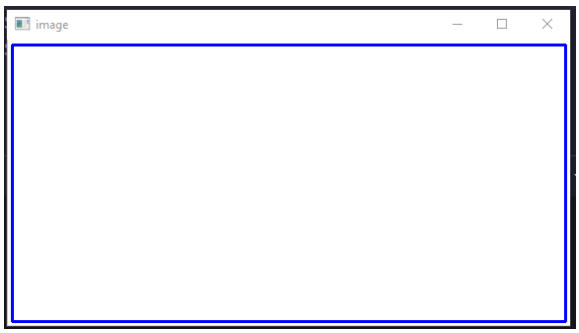
تغییر سایز عکس :

```
## resizing image
WIDTH = 290
HEIGHT = 570

img_resized = cv2.resize(image_rgb , (HEIGHT , WIDTH))
```

به کمک کد های زیر مستطیل را به کمک 4 خط رسم می کنیم :

```
## draw lines
COLOR = (255, 0, 0)
THICKNESS = 2

image = cv2.line(img_resized, (5,5), (565,5), COLOR, THICKNESS)
image = cv2.line(img_resized, (5,5), (5,285), COLOR, THICKNESS)
image = cv2.line(img_resized, (565,285), (565,5), COLOR, THICKNESS)
image = cv2.line(img_resized, (565,285), (5,285), COLOR, THICKNESS)
show_image(image=image)
```

چون طول مستطیل از طول عکس 10 تا کمتر بود نقطه شروع و پایان هم در فاصله 5 تایی از چپ و راست عکس هستند ، همینطور برای عرض . نتیجه به شکل رو به رو شد :



توسط قطعه کد زیر چهاردایره در چهارگوشه مستطیل رسم میکنیم :

```
image = cv2.circle(image , (5,5),5 , (0,0,255) , THICKNESS)
image = cv2.circle(image , (565,5),5 , (0,0,255) , THICKNESS)
image = cv2.circle(image , (5,285),5 , (0,0,255) , THICKNESS)
image = cv2.circle(image , (565,285),5 , (0,0,255) , THICKNESS)
show_image(image=image)
```

سپس به کمک حلقه for کد را ساده تر کردم :

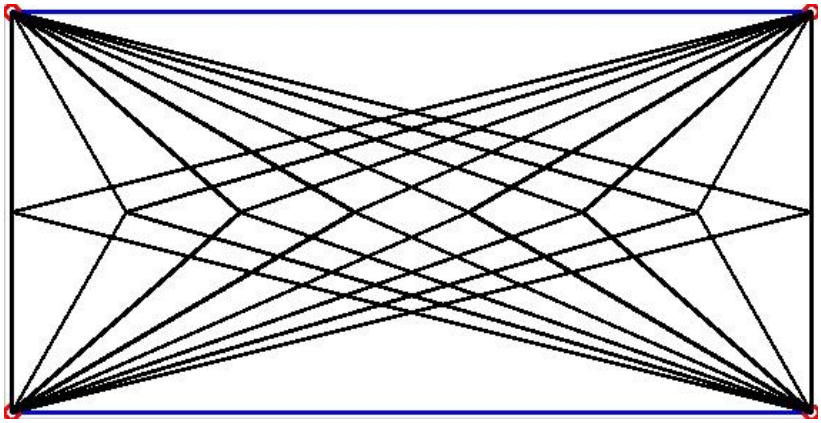
```
corners = [
    (5,5),
    (565,5),
    (5,285),
    (565,285),
]

for corner in corners :
    image = cv2.circle(image , corner,5 , (0,0,255) , THICKNESS)
```

که نتیجه به شکل زیر شد :

حال میخواهیم تمام شکل end.png را رسم کنیم . در این عکس، از چهارگوشه به 8 نقطه که در محور y ها در وسط قرار دارند و فاصله یکسانی از هم دارند وصل شده اند . از آنجایی که طول مستطیل 560 است ، در هر 80 واحد طولی ، یک نقطه گذاشته می شود . کد نهایی به شکل زیر شد :

```
for i in range(8):
    x_point = i * 80 + 5
    for corner in corners :
        end = cv2.line(img_resized,corner , (x_point,145), COLOR, THICKNESS)
```

و عکس نهایی به شکل زیر:



عکس را توسط متد imshow نمایش می دهیم . فقط مشکلی که با این متد داشتم این بود که پنجره عکس سریع بسته می شد ، برای همین از دو متد دیگر استفاده کردم که در شکل زیر می توانید ببینید :

```
def show_image(image):
    cv2.imshow('image' , image)
    cv2.waitKey()
    cv2.destroyAllWindows()
```

منبع استفاده از این دو متد جدید :

[Python Opencv2 imshow is closing immediately even with waitKey(0) - Stack Overflow]

توسط خط زیر عکس را ذخیره میکنیم (فایل عکس ضمیمه شده است) :

```
## save picture
cv2.imwrite('mypic.jpg',image)
```