به نام خدا

تمرین اول – بینایی کامپیوتر **نیایش خانی ۹۹۵۲۱۲۳۵**

سوال ۱) اگر یک تصویر را به ۶۴ سطح کوانتیزه کنیم، محدوده جدید روشنایی پیکسل ها ۶۴ (از ۰ تا ۶۳) خواهد شد. اگر مقدار ۲۵۶ را به ۶۴ تقسیم کنیم، داریم :

$$\frac{256}{64} = 4$$

بنابراین برای بدست آوردن مقدار جدید هر پیکسل باید آن را به ۴ تقسیم کنیم و از آنجایی که مقدار روشنایی یک پیکسل باید integer باشد، آن را به نزدیکترین مقدار گرد می کنیم. مثلا مقدار پیکسل ۲۴۹ به ۶۲ تغییر پیدا می کند.

این کاهش تعداد سطوح می تواند منجر به از دست رفتن جزئیات، کاهش کنتراست و smoothness می شود زیرا تعداد مقادیر gray scale متمایز را کاهش می دهد. این اتفاق مخصوصا در در مناطقی از تصویر که تغییرات ظریفی در در روشنایی دارند رخ می دهد.

سوال ۲) همانطور که می دانیم شاتر دریچه ای است که هنگام عکس برداری برای لحظه ای کنار می رود تا حسگر انرژی را دریافت کند. برای ثبت تصویری واضح از صحنه هایی که سریع رخ می دهند نیاز به شاتری سریع داریم زیرا هرچه سرعت شاتر بیشتر باشد، زمان کمتری برای نوردهی سنسور فراهم می شود. وقتی سنسور برای مدت کوتاه تری نوردهی می شود، هر حرکتی که در این زمان رخ می دهد، کمتر احتمال دارد که ثبت شود. بنابراین استفاده از یک شاتر سریع تر باعث می شود که motion blur (مات شدگی به دلیل حرکت) کمتر شود و تصویری واضح تر و تیزتر(sharp) ثبت کنیم.

البته مقدار دقیق این سرعت به عواملی مانند سرعت پرنده، فاصله دوربین تا پرنده و فاصله کانونی لنز بستگی دارد. با توجه به منبع ذکر شده سرعت ۱/۲۵۰۰ تا ۱/۳۲۰۰ و حتی بیشتر (اگر نور اجازه می دهد) مناسب است.

منبع : https://www.allaboutbirds.org/news/how-to-photograph-birds-in-flight

سوال ۳) از آنجایی که صحنه پر سرعت است، مطابق سوال قبل، به دوربینی با سرعت شاتر بالا نیازمندیم تا تصویری واضح و بدون تار ثبت کنیم. برای داشتن فوکوس روی یوزپلنگ و تار بودن پس زمینه باید DOF (عمق میدان) را تنظیم کنیم. طبق منبع برای ثبت تصاویری که در یک نقطه متمرکز و در نقاط دیگر تار هستند، عمق میدان *shallow* مناسب تر است. پس برای فوکوس روی یوزپلنگ و تر بودن پس زمینه نیاز به DOF کمتری داریم. همچنین برای پوشش دادن فضای زیادی از منظره نیاز به میدان دید بزرگ تری نسبت به قبل داریم و همانطور که می دانیم میدان دید به فاصله کانونی لنز بستگی دارد ؛ هرچه فاصله کانونی کمتر باشد، میدان دید وسیع تری خواهیم داشت. بنابراین دوربینی با فاصله کانونی کمتر برای ثبت تصویر ما مناسب تر است.

منبع : https://boords.com/blog/what-is-shallow-depth-of-field-definition-and-examples#:~:text=A%20shallow%20depth%20of%20field%20describes%20a%20small%20area%20or,%2C%20distance%2C%20and%20focal%20length.

سوال ۵)

الف) همانطور که مشاهده می شود تصویر نمایش داده شده توسط این دو متود با هم تفاوت رنگی دارند که این تفاوت به دلیل این است که به طور پیش‌فرض، OpenCV تصاویر را با فرمت BGR (آبی، سبز، قرمز) می‌خواند، در حالی که Matplotlib فرض می‌کند که تصاویر در فرمت RGB (قرمز، سبز، آبی) هستند. این بدان معنی است که هنگام رندر کردن یک تصویر با استفاده از OpenCV، کانال های رنگی متفاوت از Matplotlib تفسیر می شوند و در نتیجه تفاوت هایی در نمایش رنگ ایجاد می شود.

همچنین Matplotlib به طور خودکار انجام پیکسل‌های تصویر را به محدوده [۰، ۱] نرمالایز می‌کند در حالی که OpenCV این کار را به صورت پیشفرض نمی‌دهد. این می‌تواند بر روی روشنایی و کنتراست تصویر تأثیر بگذارد.

ب) ابتدا باید خروجی Matplotlib را مانند OpenCV بسازیم. این کار را با قطعه کد زیر انجام می دهیم.

```
image = cv2.imread('ComputerDepartment.jpg')

# Convert the image from BGR to RGB
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

plt.imshow(image_rgb)
plt.axis('off')
plt.show()
```

این قطعه کد یک تصویر را با استفاده از OpenCV می خواند، آن را از فرمت BGR به RGB تبدیل می کند و سپس آن را با استفاده از Matplotlib بدون نرمال سازی نمایش می دهد. با انجام این تبدیل، اطمینان حاصل می کنیم که کانال‌های رنگی به درستی تفسیر شده‌اند. علاوه بر این، plt.axis('off') محور های مختصات و ... که در نمایش matplotlib استفاده می‌شود را حذف می کند.

ب-۱) برای رسیدن به این تصویر از کد زیر استفاده شده است.

```
image_flipped = np.flipud(image_rgb)

# Concatenate the original image with its flipped version vertically
combined_image = np.concatenate((image_rgb, image_flipped), axis=0)
```

تابع np.flipud یک آرایه را در در محور عمودی خود منعکس می کند. بنابراین عکس ورودی این تابع به صورت معکوس (از بالا به پایین) تبدیل می شود.

سپس در خط بعدی به تابع np.concatenate عکس اصلی و عکس معکوس شده را می دهیم تا این دو را به صورت عمودی (axis=0) به هم متصل کند. نتیجه یک تصویر جدید است که شامل تصویر اصلی در بالا و تصویر معکوس در پایین است.

ب-۲) تابع np.concatentenate به صورت دیفالت ورودی اول بالا و ورودی دوم را در پایین قرار می دهد بنابراین برای رسیدن به تصویر مطلوب تصویر معکوس شده در قسمت قبل را به ورودی اول و تصویر اصلی را به ورودی دوم می دهیم.

ب-۳)

```
image_flipped_horizontal = np.fliplr(image_rgb)
combined_image_horizontal = np.concatenate((image_rgb, image_flipped_horizontal),
axis=1)
```

تابع np.fliplr تصویر را به صورت افقی معکوس می کند. سپس تصویر حاصل را به تابع np.concatenate می دهیم تا دو تصویر را به هم متصل کند. اتصال افقی این دو تصویر با axis=1 حاصل می شود.

ب-۴) همانطور که در قسمت ب-۲ توضیح داده شد، با تغییر ترتیب ورودی ها به تصویر مطلوب خواهیم رسید.

ب-۵)

```
# Read the image with OpenCV
image = cv2.imread('ComputerDepartment.jpg')

# Define the coordinates of the rectangle
top_left = (168,225)
bottom_right = (145, 197)

#Set the color and thickness of the rectangle
rectangle_color = (0, 255, 255)
thickness = cv2.FILLED

# Draw the rectangle on the image
cv2.rectangle(image, top_left, bottom_right, rectangle_color, thickness)

image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

plt.imshow(image_rgb)
plt.axis('off')
# plt.show()
```

در این قطعه کد بعد از خواندن تصویر مختصات بالا سمت چپ و پایین سمت راست مستطیل را تعریف می کنیم. (اعداد با صحیح و خطا به دست آمده اند تا بر روی یکی از پنجره ها منطبق شوند.) در دو خط پایین تر رنگ و میزان ضخامت مستطیل مشخص شده است که رنگ آن زرد و ضخامت آن به صورت filled قرار گرفته است تا مستطیلی توپر داشته باشیم.

تابع rectangle یک مستطیل را روی عکس ورودی، با مختصات، رنگ و ضخامت مشخص شده رسم می کند. همانطور که در ابتدای سوال ۵ دانستیم که matplotlib و opencv از نظر رنگ با هم تفاوت دارند، همان کد اینجا نیز استفاده شده است که تصویر را به فضای رنگی RGB ببریم.

برای ذخیره سازی تصویر از کد زیر استفاده می کنیم.

```
plt.savefig('Colored_Window.jpg', bbox_inches='tight', pad_inches=0)
```

این خط تصویری که در حال حاضر در حافظه توسط matplotlib ایجاد شده است را با نامی که در پارامتر اول داده ایم، ذخیره می کند پارامتر بعدی باعث می شود که حاشیه های اضافی از تصویر حذف شوند و پارامتر آخر باعث می شود که هیچ فضای اضافی در اطراف تصویر وجود نداشته باشد.

سوال ۶) بعد از نمایش تصویر به سراغ نمایش کانال ها می رویم. برای نمایش کانال آبی از کد زیر استفاده شده است:

```
blue_channel = logo_rgb.copy()
blue_channel[:, :, 0] = 0  # Set red channel to zero
blue_channel[:, :, 1] = 0  # Set green channel to zero
```

همانطور که می دانیم رنگ ها در یک آرایه سه بعدی ذخیره می شوند که بعد سوم متعلق به کانال های رنگی است. در فضای *RGB* ایندکس صفر قرمز، ایندکس یک سبز و ایندکس دو آبی است. بنابراین برای اعمال کانال آبی روی تصویر، یک کپی از تصویر می گیریم تا تغییرات را روی آن اعمال کنیم سپس ایندکس صفرم بُعد سوم آرایه را صفر قرار می دهیم تا کانال قرمز را صفر کنیم. سپس همین کار را برای ایندکس اول انجام می دهیم تا کانال سبز هم صفر شود.

همین کار را برای رسیدن به کانال های دیگر نیز انجام می دهیم. برای کانال سبز ایندکس صفر (کانال قرمز) و ایندکس ۲ (کانال آبی) و برای کانال قرمز ایندکس یک و دو (کانال های سبز و آبی) را صفر قرار می دهیم.

تحلیل اعمال کانال ها: ابتدا با اعمال کانال آبی روی تصویر می بینیم که نقاط آبی رنگ به همراه نقاط سفید رنگ همگی به رنگ آبی در می آیند زیرا نقاطی که شدت رنگ آبی در آنها غیر صفر است، به رنگ آبی در می آیند. در مورد یک پیکسل سفید، هر سه کانال رنگی (قرمز، سبز و آبی) دارای حداکثر شدت هستند و در نتیجه پیکسلی سفید به نظر می رسد. بنابراین در این نقاط، نمایش یک کانال رنگی خاص باعث می‌شود که آن نواحی با رنگ کانال نمایش داده شده ظاهر شوند. باقی نقاط که در آنها شدت رنگ آبی صفر است، پیکسل آنها به رنگ مشکی نمایان می شود. بنابراین مشعل موجود در لوگو و متن "دانشگاه علم و صنعت ایران " به رنگ مشکی و دسته مشعل و چرخ دنده به همراه بک گراند تصویر به رنگ آبی دیده می‌شوند.

در کانال قرمز نیز به ترتیب قبل، مشعل و بک گراند تصویر به رنگ قرمز و باقی نقاط که آبی و مشکی هستند به رنگ مشکی ظاهر می شوند. همچنین در کانال سبز به دلیل وجود نداشتن رنگ سبز در تصویر اصلی، پیکسل های سفید به رنگ سبز و باقی پیکسل ها به رنگ مشکی ظاهر می شوند. در کل نتیجه می گیریم که رنگ سفید و رنگی که مشابه کانال است به رنگ کانال و رنگی غیر مشابه با رنگ کانال به رنگ سیاه تبدیل می شود.