دانشگاه علم و صنعت

تمرین اول مبانی بینایی کامپیوتر

نام و نام خانوادگی:

فرناز خوش دوست آزاد

شماره دانشجویی:

99521253

نام استاد:

دکتر محمدرضا محمدی

1

از آنجا که 256، 4 برابر 64 می باشد، این بدین معناست که ما باید مقادیر را به ترتیب 4 تا 4 تا به یک مقدار نسبت دهیم که به شرح زیر است:

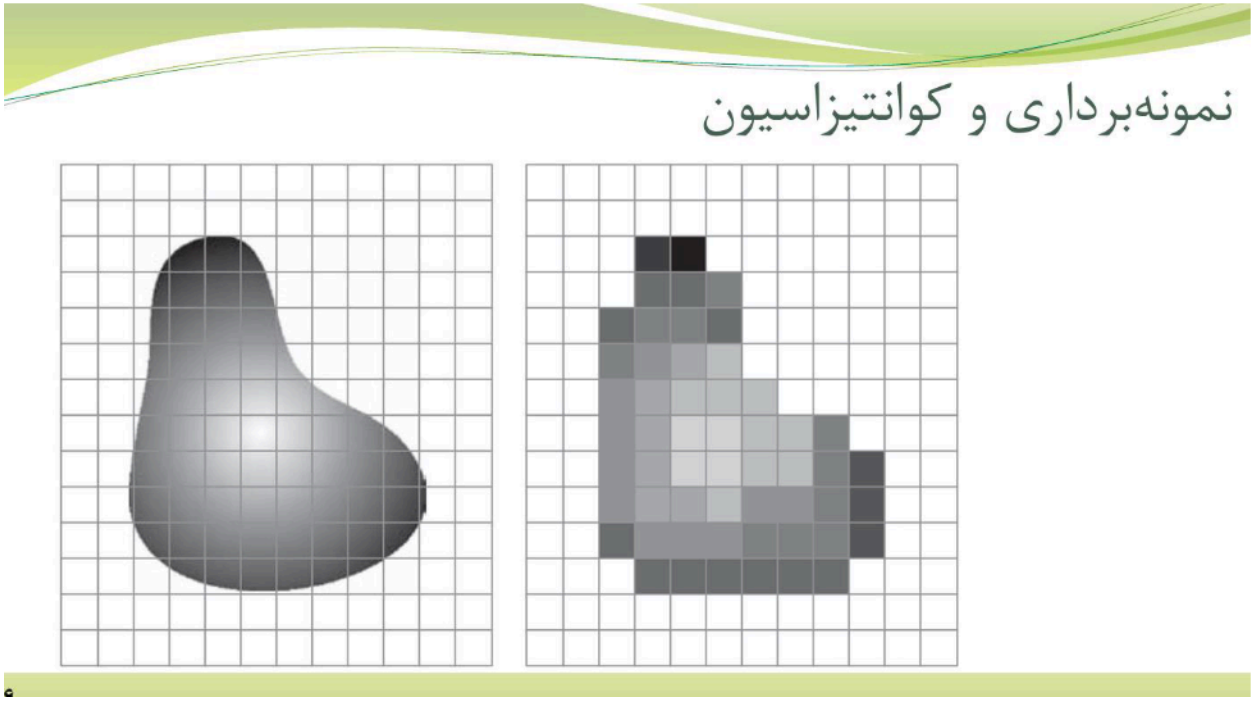
0 - 3→ 0

4 - 7→ 1

8 - 11 → 2

...

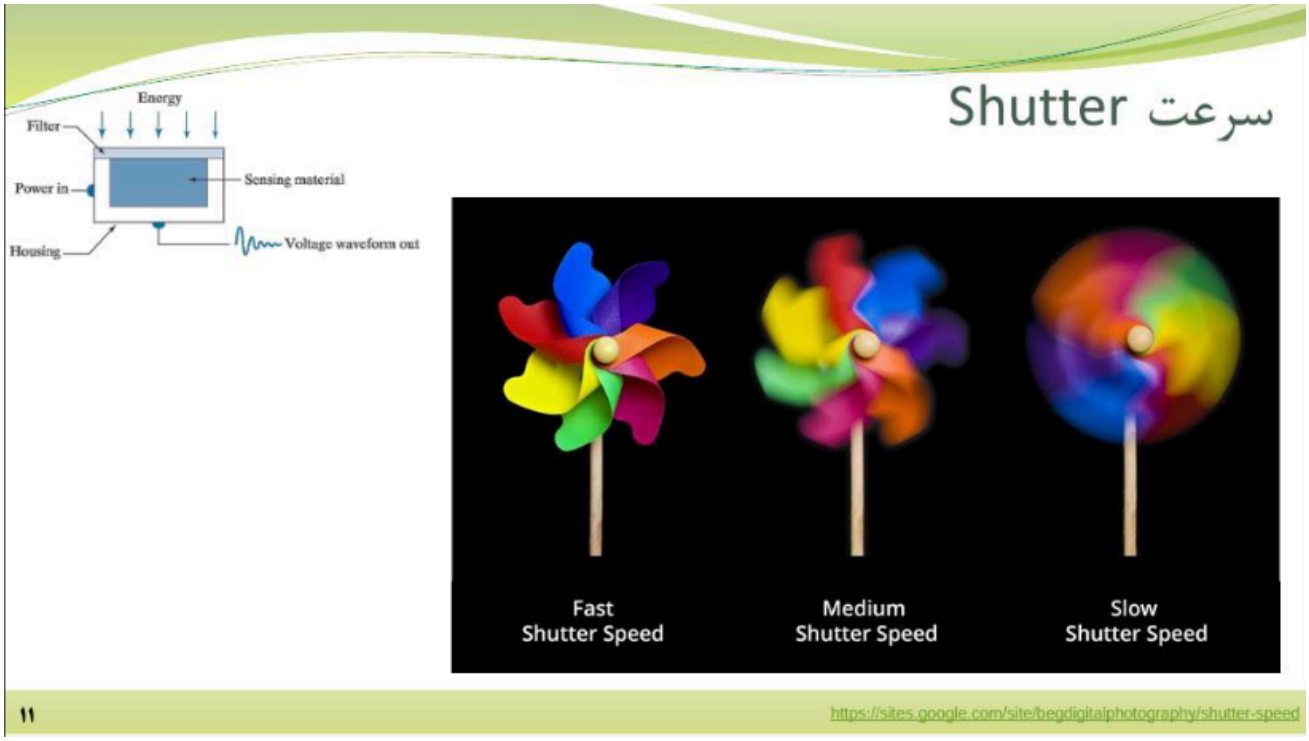
که این یعنی اطلاعات بسیاری را از دست می دهیم و تعداد سطوح کم می شود و حجم عکس کمتر می شود و از طرفی transition سطوح مشخص تر می شود که همانطور که در عکس زیر نیز دیده می شود، پیکسل های پرجزئیات و پراطلاعات(گوشه های تیز و تغییرات فرکانس بالا) نسبت به عکس های با تغییرات نرم در روشنایی، در مقابل این quantizqation تاثیر بیشتری می گیرند.



2

برای گرفتن عکس پرنده ی در حال حرکت بهتر است که سرعت شاتر بالا باشد، زیرا در صورتی که سرعت شاتر پایین باشد و دریچه مدت زمان زیادی باز باشد، جزئیات را نمی توان در عکس ذخیره کرد

و مانند عکس زیر که فرفره ی در حال حرکت را نشان می دهد تصویری غیر شفاف و تار و یا کشیده خواهیم داشت و جزئیات از بین خواهند رفت.



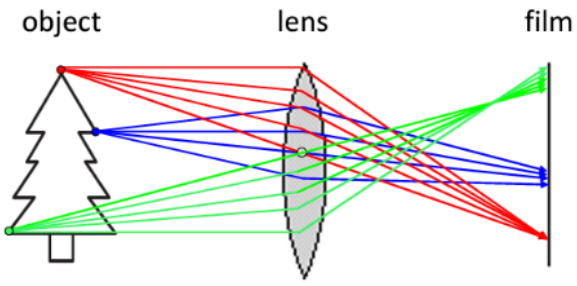
3

با توجه به اینکه سرعت یوزپلنگ بسیار بالاست پس دریچه باید کمتر باز بماند و سرعت shutter بسیار بالا باشد تا تصویر یوزپلنگ تار نباشد. با استفاده از تکنیک پنینگ (Panning Technique) نیز می توانیم برای ایجاد حس سرعت و همچنین فوکوس روی یوزپلنگ با پس‌زمینه‌ای تار، عکاس می‌تواند از تکنیک پنینگ استفاده کرد. در این تکنیک، عکاس دوربین را همراه با حرکت سوژه حرکت می‌دهد و با سرعت shutter نسبتاً پایین‌تری عکس می‌گیرد. این کار باعث می‌شود که سوژه نسبتاً واضح بماند در حالی که پس‌زمینه تاریکی حرکت ایجاد می‌کند.

برای ایجاد پس‌زمینه‌ای تار و فوکوس روی یوزپلنگ، عکاس باید از یک دیافراگم باز (کمترین عدد F ممکن) استفاده کند. این کار باعث کاهش عمق میدان می‌شود و فقط سوژه در فوکوس باقی می‌ماند، در حالی که پس‌زمینه تار می‌شود وطبق فرمول 1 /1/u + 1/v = fاز آنجایی که فاصله کانونی ثابت است با زیاد کردن فاصله لنز تا فیلم dof ما به لنز نزدیکتر شده و یوزپلنگ با فوکوس خوب و پس زمینه تار ثبت خواهد شد.

معادلات لنز نازک

- تنها اشعه‌های نوری نقاطی که در فاصله u از لنز باشند در صفحه‌ای به فاصله v از لنز همگرا (متمرکز) می‌شوند
- نقاط با فاصله‌های دیگر دچار تاری خواهند شد



$$\frac{1}{f}=\frac{1}{v}+\frac{1}{u}$$

۴

4

فایل های مربوطه ران و خوانده شده اند.

5

الف)یکی از تفاوت‌های اصلی بین نمایش تصویر با کتابخانه‌های OpenCV و Matplotlib در نحوه تفسیر رنگ‌ها است. OpenCV تصاویر را در فرمت BGR (آبی، سبز، قرمز) می‌خواند و ذخیره می‌کند، در حالی که Matplotlib تصاویر را در فرمت RGB (قرمز، سبز، آبی) نمایش می‌دهد. این تفاوت در فرمت رنگ می‌تواند باعث شود تصاویری که با OpenCV خوانده شده‌اند، هنگام نمایش در

Matplotlib، رنگ‌های غیرمنتظره‌ای داشته باشند. برای مثال، تصویری که در OpenCV آبی به نظر می‌رسد، ممکن است هنگام نمایش در Matplotlib به رنگ قرمز ظاهر شود. که برای حل این چالش از متود cvtColor در cv2 استفاده می کنیم.

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

ب) تابع flip عکس ها را با توجه به پارامتر دوم که axis یا همان محور مختصات است اگر صفر بدیم عمودی و اگر یک بدیم افقی آن را معکوس میکند. تابع concatenate هم در واقع قرار دادن دو آرایه numpy زیر یکدیگر را بر عهده دارد(با توجه به تناظر بعدها با یکدیگر). برای پیدا کردن پنجره نیز یکی را با امتحان کردن مختصات های مختلف پیدا و رنگی می کنیم و برای رنگ زرد هم باید رنگ سبز و قرمز را 255 و آبی را 0 کنیم.

Mode1:

Mode 1 : The main image should be joined vertically with its vertical invert.

```
1  ##########################Start Code#####################################
2  import tensorflow as tf
3  import matplotlib.pyplot as plt
4  image_path = "ComputerDepartment.jpg"
5  image_string = tf.io.read_file(image_path)
6  image = tf.image.decode_jpeg(image_string, channels=3)
7  flipped_image = tf.image.flip_up_down(image)
8  concatenated_image = tf.concat([image, flipped_image], axis=0)
9  plt.figure(figsize=(10, 20))  # Adjust the figure size as needed
10 plt.imshow(concatenated_image)
11 plt.axis('off')  # Hide the axis
12 plt.show()
13 ##########################End Code#####################################
```

Mode2:

Mode 2 : The vertical invert of the original image should be joined vertically with the original image.

```
[23]  1  ##########################Start Code#####################################
      2  import tensorflow as tf
      3  import matplotlib.pyplot as plt
      4  image_path = "ComputerDepartment.jpg"
      5  image_string = tf.io.read_file(image_path)
      6  image = tf.image.decode_jpeg(image_string, channels=3)
      7  flipped_image = tf.image.flip_up_down(image)
      8  concatenated_image = tf.concat([ flipped_image, image], axis=0)
      9  plt.figure(figsize=(10, 20))  # Adjust the figure size as needed
     10  plt.imshow(concatenated_image)
     11  plt.axis('off')  # Hide the axis
     12  plt.show()
     13  ############################End Code#####################################
```

Mode3:

Mode 3 : The main image should be joined horizontally with its horizontal invert.

```
 1  ##########################Start Code#####################################
 2  image_path = "ComputerDepartment.jpg"
 3  image = tf.io.read_file(image_path)
 4  image = tf.image.decode_jpeg(image, channels=3)
 5  image = tf.image.convert_image_dtype(image, tf.float32)
 6  flipped_image = tf.image.flip_left_right(image)
 7  concatenated_image = tf.concat([image, flipped_image], axis=1)
 8  plt.figure(figsize=(10, 5))
 9  plt.imshow(concatenated_image)
10  plt.axis('off')
11  plt.show()
12  ############################End Code#####################################
```

Mode4:

Mode 4 : The horizontal invert of the original image should be joined horizontally with the original image.

```
##########################Start Code#####################################
image_path = "ComputerDepartment.jpg"
image = tf.io.read_file(image_path)
image = tf.image.decode_jpeg(image, channels=3)
image = tf.image.convert_image_dtype(image, tf.float32)
flipped_image = tf.image.flip_left_right(image)
concatenated_image = tf.concat([flipped_image, image], axis=1)
plt.figure(figsize=(10, 5))
plt.imshow(concatenated_image)
plt.axis('off')
plt.show()
##########################End Code#####################################
```

Mode5:

Change the color of one of the windows of the computer faculty building as desired. Try to do this very carefully.

```
##########################Start Code#####################################
img2 = cv2.imread("ComputerDepartment.jpg")
test = img2
test[310:360, 275:300] = (255,255,0)
plt.imshow(test)
##########################End Code#####################################
```

6

در ابتدا عکس ها را خواندیم و آنها را rgb کردیم بعد عدد مربوط به هر بخش کانال دیگر را صفر کردیم. مثال برای نمایش کانال قرمز، کانال سبز و کانال آبی را صفر کردیم و عکس را نمایش دادیم و برای کانال سبز و آبی نیز همین کار را تکرار کردیم و با توجه به اینکه رنگ سفید از هر سه رنگ تشکیل شده است پس در هر سه کانال مقدار دارد و عکس به همان صورتی که نشان داده شده است، عوض شده است چون آبی است پس باقی رنگ ها به جز آبی مقدار صفر دارد پس سیاه می شود. برای شعله نیز فقط کانال قرمز رنگ مکس 255 دارد و باقی صفر هستند پس سیاه خواهند بود.

Show blue channel.

```
[29]
##########################Start Code#####################################
B, G, R = cv2.split(logo)
blue_channel_img = cv2.merge([B, np.zeros(B.shape, dtype=B.dtype), np.zeros(B.shape, dtype=B.dtype)])
cv2_imshow(blue_channel_img)
cv2.waitKey(0)
cv2.destroyAllWindows()
##########################End Code#####################################
```

کد بالا مربوط به کانال آبی می باشد و برای کانال های آبی و سبز نیز همین کار را با مقادیر ویژه ی آنها پر می کنیم.