بسم الله الرحمن الرحیم

محمد عرفان زارع زردینی

۹۸۴۱۱۱۴۳۲

تمرین سری دوم درس بینایی ماشین

1)

Refrences:

https://www.geeksforgeeks.org/difference-between-rgb-cmyk-hsv-and-yiq-color-models/

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

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

الف)

در حل قسمت اول باتوجه به فرمول در تصویر ، کد مربوطه را مینویسیم.برای یافتن مقدار برعکس آن نیز فرمول نوشته شده را پیاده سازی می نماییم.خروجی cmyk به درصد و در 100 ضرب شده است.

RGB to CMYK conversion formula

The R,G,B values are divided by 255 to change the range from 0..255 to 0..1:

$$R' = R/255$$

$$G' = G/255$$

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

The black key (K) color is calculated from the red (R'), green (G') and blue (B') colors:

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

The cyan color (C) is calculated from the red (R') and black (K) colors:

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

The magenta color (M) is calculated from the green (G') and black (K) colors:

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

The yellow color (Y) is calculated from the blue (B') and black (K) colors:

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

CMYK to RGB conversion formula

The *R,G,B* values are given in the range of 0..255.

The red (R) color is calculated from the cyan (C) and black (K) colors:

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

The green color (G) is calculated from the magenta (M) and black (K) colors:

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

The blue color (B) is calculated from the yellow (Y) and black (K) colors:

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

ب)

برای تبدیل عکس خوانده شده و بردنش به فرمت ycbr وhsv ،کافی است تصویر را با cv2 که bgrمیخواند بخونیم و با cvtcolorو دادن فرمت cv2.COLOR_BGR2YCR_CB و cv2.COLOR_BGR2YCR_HSV ، خواسته سوال را برطرف کرده و تصاویر مدنظر را چاپ می نماییم.

ج)

برای جدا کردن کانال های تصویر در فرمت hsv ،کافی است با تابع آماده توضیح داده شده در قسمت قبل و جدا سازی کانال و ذخیره هر بخش به صورت جدا جدا ، مقادیر مدنظر را چاپ می کنیم.

د)

برای یافتن تفاوت دوتصویر کافی است، ابتدا تصاویر را هم سایز کرده ، سپس یک متغیر از نوع ndarray میسازیم که با توجه به فرمت عکس، نوع uint8 را نوع متغیر تعریف و آن را 3 کاناله می سازیم. حال می ایم برای تعیین تفاوت و یافتن جواب مدنظر، برای ساخت عکس جدید، از یک عکس یک کانال را خوانده و دوکانال دیگرش را از عکس دیگر می اوریم. تا بدین سان با توجه به assignشدن و اینکه اگر تفاوت درعکس ها باشد ، با assign شدن این فرمت عکس برای کانال، به جواب دلخواه و مدنظر می رسیم. در واقع بدین سان اگر منطبق باشند آن قسمت سیاه سفید میشه و در غیر آن صورت قرمز یا فیروزه ای می شوند.

ه)

از دلایل آن می توان نکات زیر را اشاره کرد:

1- نمایش اطلاعات رنگ: فضاهای رنگی راه های مختلفی برای نمایش اطلاعات رنگ ارائه می دهد. برای مثال فضای rgb شامل سه رنگ قرمز و سبز و آبی اند در حالی که فضای hsv اطلاعات رنگ را با مولفه رنگ، اشباع و نور رنگی نمایش می دهد(این فضا به نحو و دزک وشناخت انسان نزدیکتر است و در گرافیک کاربرد دارد). برحسب نیاز و کاربرد، یک فضای رنگی برای نمایش اطلاعات رنگی مناسب تر از دیگری است.

2- عملیات پردازش تصویر: فضای رنگی می تواند برای برخی از عملیات پردازش تصویر مناسب تر باشد .مثال ما یک فضای رنگی به نام yuv معمولا برای فشرده سازی ویدئو استفاده می شود ، زیرا اطلاعات درخشندگی(روشنایی) و رنگ یک تصویر را از هم جدا می نماید که امکان فشرده سازی کارآمد اطلاعات را فراهم می نماید.

3- درک رنگ: اسنا ها در شرایط نوری و محیط های مختلف رنگ را به شکل متفاوتی درک می کند.پس می شود فضای رنگی را به شکل طراحی نمود که تفاوت ها را در نظر گرفت و اطلاعات دقیقی را نمایش داد.

4- سازگاری با نرم افزار وسخت افزار: دستگاه های نرم افزاری و سخت افزاری ممکن است که از فضاهای رنگی متفاوتی استفاده نماید. با استفاده از فضاهای رنگی متعدد ، سیستم های بینایی کامپیوتر میتواند سازگاری با دستگاه ها وبرنامه های متفاوت را تضمین نماید.مثال مانیتور ها با خاموش روشن کردن چراغ های ریز rgb نمایش می دهند. در حالیکه جاپگر ها قادر به زیاد کردن نور کاغذنبوده وکاغذ سفید را دریافت و با رنگ های فیروزه ای، بنفش و زرد را قراردادده تا rgb مدنظر و رنگ ها را تولید نماید.

به شکل کلی فضاهای رنگی متعدد، انعطاف پذیری را فراهم نموده و امکان نمایش و پردازش دقیق تر اطلاعات رنگ را برای کاربرد های مختلف فراهم نمود.

(2

Refrences:

[OpenCV Panorama Stitching - GeeksforGeeks]

برای ساخت تصویر، ابتدا آدرس عکس ها را در یک آرایه می نویسیم .سپس عکس ها را به شکل جدا ازهم خوانده و نمایش می دهیم. حال با استفاده از تابع cv2.stitcherاستفاده مینماییم که ابتدا آن را ساخته، سپس با stitch()() که متعلق به کلاس stitcher در ماژول opencv است، ارایه ای از تصویر را به عنوان ورودی گرفته و آن ها را به هم میچسباند .تابع ،تصویر حاصل و یک مقدار bool برمی گرداند که اگر چسباندن عکس ها درست انجام شد true و درغیر این صورت false برمیگرداند. در صورت درست انجام شدن، خروجی را چاپ می نماییم.

3)

Refrences:

https://pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/

.....

در تابع put_mask که ورودی آن چهره فرد و تصویر ماسک است؛ قرار بر این است ماسک به شکلی تنظیم شود که روی صورت فرد به درستی قرار گیرد.ابتدا توابع تشخیص چهره را مینویسیم که get_frontal_face_detector می باشد.تابع پیش بینی شکل به مدلی از تبدیل داده نیاز دارد که در لینک زیر موجود است.

https://raw.githubusercontent.com/tzutalin/dlib-android/master/data/shape_predictor_68_face_landmarks.dat

سپس مستطیل حاوی چهره بوسیله تابع تشخیص بدست آمده و مختصات آن همراه با تصویر چهرهبه تابع پیش بینی داده می شود.خروجی کار لیستی از 68نقطه کلیدی می باشد.

من نقاط 2و5و13و16 را به عنوان نقطه مقصد برای تابع تبدیل گرفتم.

توابع تشخیص چهره قادر به تشخیص ماسک نبوده ، پس نقاط کلیدی را به شکل دستی مشخص و به عناون نقطه مبدا در نظر گرفتیم و سپس براساس نقاط مبدا scale دادیم.

سپس ماتریس نمایگر تابع تبدیلمان را با استفاده از تابع getPerspectiveTransform با ورودی های مبدا و مقصد نمایش دادیم.سپس ماسک را بوسیله ماتریس یافته شده وتابع warpPerspective ، و در نهایت بکگراند ماسک تبدیلی را یافته و کانال الفا را به تصویر ماسک و چهره افزوده و با بکگراند، دو تصویر را با هم ترکیب می نماییم.

4)

Refrences:

https://stackoverflow.com/questions/8535650/how-to-change-saturation-values-with-opencv

الف)

ابتدا عکس را میخوانیم و با توجه به نویز داشتن عکس با فیلتر گاوسی تلاش به کم کردن آن می نماییم. پارامتر ها با سعی و خطا و مناسب سازی بذست می آید.سپس عکس را به فضای grayscaleمیبریم.سپس با لبه یاب کنی و سعی وخطا مقدار مناسب برای ورودی هایش را مشخص می نماییم.

ب)

حال با تابع findcontours که روی خروجی قبلی زده می شود ، contours ها را می یابیم.

ج)

در اینجا باید از میان contourها اون هایی که در گوشه اند را بیابیم. که با دو حلقه تودرتو مختصات دو کانتوری که از هم بیشترین فاصله را دارند را یافته و به عنوان گوشه در نظر میگیریم.

حال با توابع perpective transformکار داریم برای رساندن ابعاد کاغذ به مقدار اصلی.

از دوتابع نوشته شده در کد استفاده می نماییم که getPerspectiveTransform و warpPerspective هستند.که به اولی دو سری نقطه ورودی داده می شود تا ماتریس تبدیل ساخته شود که نقاط اولیه نقاط گوشه اند ونقاط بخش دوم 4 نقطه گوشه اند. در نهایت هم ماتریس تبدیل به همراه نقاط گوشه را به تابع دوم داده تا بدین سان تصویر را تبدیل کرده و خروجی مناسب را چاپ کند .در نهایت هم همه تصاویر و مراحل را نشان می دهیم.

د)

برای بهبود تصویر از افزایش غلظت تصویر(Saturation) استفاده نمودیم. برای این کار ابتدا فضای رنگی تصویر را به HSV تغییر دادم و بعد از ۳برابر کردن مقدار S تصویر را به فضای رنگی پیش فرض BGR برگرداندم. در ران کردن کد به علت تغییر کانال ها ، به ارور خورده است خروجی تست ولی بیس کد درست می باشد.

5)

Refrences:

https://www.geeksforgeeks.org/erosion-dilation-images-using-opencv-python/

الف)

Harris یک تصویر را ورودی گرفته و (اغلب با استفاده از sobel) گرادیان محور های x,y را یافته و مربع و حاصل ضرب دو مشتق را حساب و هرکدام را از یک فیلتر(بیشتر گاوسی) عبور و حاصل را ذخیره مینمایید.در فرمول زیر w(x,y) اثر فیلتر است.)

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

سپس برای هر نقطه مان مقدار R را براساس ماتریس متناظر میابیم. در فرمول det(m) دترمینان m هست و trace(m) حاصل جمع اعضای قطر اصلی m بوده و k پارامتر واسط است.

ب)

مانند قسمت الف تابع مورد نیاز را پیش برده وپیاده سازی می نماییم. در اینجا فقط ابتدا تصویر را سیاه سفید کرده و نوع پیکسل آن را تغییر می دهیم. نقاط سفید گوشه های تصویر و نقاط سیاه هم لبه اند.

برای قسمت استفاده شده از توابع آماده ، با دستور cv2.cornerharrisکه پارامتر هایی مشابه تابع قبلی دستی دارد برروی تصویر زده سپس dilate رو تصویر اعمال کرده و سپس با افزودن و بررسی ضریبی ، نقاط گوشه را آبی می نماییم.سپس هم نمایش می دهیم نتیجه را.

دیده می شود که نقاط گوشه پیدا شده در نتایج هر دو خروجی به هم نزدیک اند و دقت خوبی را دارند.(در دومی عکس را برای وضوح بیشتر سیاه سفید چاپ و بررسی نکردیم.

6)

Refrences:

https://mikhail-kennerley.medium.com/a-comparison-of-sift-surf-and-orb-on-opencv-59119b9ec3d0

https://www.analyticsvidhya.com/blog/2019/10/detailed-guide-powerful-sift-technique-image-matching-python/#:~:text=SIFT%20algorithm%20helps%20locate%20the,detection%2C%20scene%20detection%2C%20etc.

https://ieeexplore.ieee.org/document/8586755

روش surf :

برای تخمین لاپلاسین گاوسی بجای اختلاف گاوسی از فیلتر جبه ای استفاده کرده و هنگام مشخص کردن جهت ها، جواب موج هایی در محدوده اطراف نقطه با اندازه متناسب با مقیاس تصویر برای دو محور عمودی و افقی محاسبه می گردد.در این روش محدوده 20s در 20s اطراف نقطه انتخاب شده که s متناسب با مقیاس بوده .سپس به قسمت های 4❁4 و پاسخ ضربه ای آنها محاسبه و کنار هم قرار میگرد تا نمایانگر آن نقطه باشند.

روش orb:

ترکیبی از دو الگوریتم fast و briefاست . (در این روش البته تغییراتی برای جلوگیری از چرخش انجام می شود .مانند اینکه ویژگی جهت در روش اول با ضرب شدت روشنایی نواحی نزدیک نقطه کلیدی با جهت آنها و میانگین گیری حاصل شده و در روش دومی ابتدا نقطه با توجه به جهت آن چرخانده شده سپس داده ها بررسی و استخراج می شوند.

)الگوریتم اول برای یافتن نقطه کلیدی ابتدا نقطه ای را کاندید و بررسی می نماید که تعداد مشخصی نقطه متصل به هم در فاصله یکسان از آن قرار دارند به شکلی که همه از آن نقطه روشنتر

یا تیره تر باشند. در الگوریتم دوم ،تست برروی نقاط اطراف نقطه کلیدی انجام می شود و نتایج به شکل باینری برای نمایش وتوصیف نقطه برمیگردد.

روش sift :

این روش برای حذف .ابستگی به مقیاس ، لاپلاسین گاوسی را بوسیله تفاوت گاوسی تخمین و محدوده نقاط بیشینه را می یابد. در این روش در هر مرحله اندازه پنجره فیلتر کوچک شده و درنتیجه نقاط بیشینه، محدودتر می شوند تا به نقاط بیشینه محلی به شکل 3❁3❁3 برسیم.

می شود با این روش محدوده های 16❁16 اطراف هر نقطه کلیدی را به قسمتهای 4❁4 تقسیم و در هر قسمت هیستوگرام جهت را یافته و کنار هم قرار داده تا به یک توصیفگر مناسب برای آن نقطه برسیم.

مقایسه:

در حالت کلی orb از نظر سرعت و دقت و کیفیت خروجی ،عملکرد بهتری دارد . همچنین در برابر چرخش و تغییر روشنایی مقاومت خوبی دارد.