به نام خدا

## عنوان

بخش اول از تکلیف اول درس پردازش تصویر رقمی

## استاد

دکتر منصوری

## دانشجو

محمدعلی مجتهدسلیمانی

۴۰۳۳۹۰۴۵۰۴

## تاریخ

۱۴۰۴/۰۲/۵

Table of Contents

# سوال ۱ : فضا رنگ‌ها

این سوال در ۳ بخش انجام شده است که بخش ب و ج که شامل موارد پیاده سازی هستند شامل گزارش کار و نتایج خروجی هستند که در همان قسمت قابل مشاهده هستند همچنین خروجی تصویری نیز تحت عنوان PART?_output قابل مشاهده است.

## بخش الف

یک فضا رنگ، به نوعی یک مدل ریاضی برای توصیف کردن رنگ‌ها با اعداد است. به طور خاص برای مدیریت کردن و مشخص کردن رنگ‌ها استفاده میشود. هر مدل برای استفاده‌ای خاص به کار میرود که در ادامه هر کدام را بررسی میکنیم:

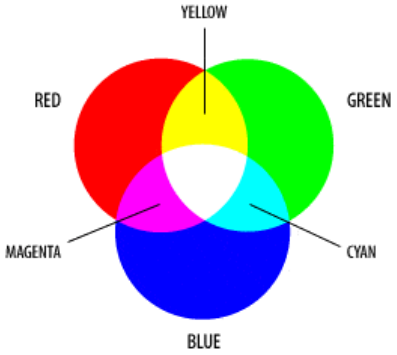
### فضا رنگ RGB

این مدل که مخفف سه رنگ آبی، سبز و قرمز هست، بر اساس نحوه ترکیب نور است که با رنگ سیاه (یعنی نبود نور) شروع میشود و با افزودن مقادیر مختلفی از نور قرمز، سبز و آبی (که رنگ های اصلی نور هستند)، طیف وسیعی از رنگ‌ها را تولید میکند.

نحوه کارکرد مدل به این شکل است که اگر این نورهای مختلف با هم ترکیب شوند رنگ های مختلفی تولید میکنند اگر هر ۳ نور با تمام شدت ترکیب شوند رنگ سفید را پدید میاورند. متغیر بودن شدت هر نور (روشنایی هر کدام) باعث تولید میلیون های رنگ جدید میشود.

هر مولفه از این فضا رنگ با یک عدد صحیح بین ۰ تا ۲۵۵ نمایش داده میشوند. ۰ به معنای عدم حضور این نور و ۲۵۵ به معنای حضور پر رنگ این نور معنا دارد. هر کدام از این موارد را نیز میتوان با ۸ بیت در هر کانال نمایش داد که اجازه میدهد ۲۵۶ مقدار ممکن برای هر مولفه داشته باشیم. در نتیجه این اتفاق ما 256*256*256*=16,777,216 رنگ ممکن خواهیم داشت. همچنین برای گزارش دقیق از هر رنگ میتوانیم از تعداد بیت‌های بیشتری برای توصیف کنیم.

این مدل در مانیتورها و دستگاه‌های هوشمند معمولا استفاده میشود که یک مدل اصلی برای دستگاه‌هایی است که نور منتشر میکنند. البته مشکلی که در این سیستم‌ها هست وابسته بودن آنها به یک دستگاه خاص است به طوری که مقدار یکسان RGB در صفحات مختلف، اندکی متفاوت هستند. همچنین این مدل میتواند طیف وسیعی از رنگ‌ها را بازنمایی کند مخصوصا رنگ‌های روشن و درخشان.

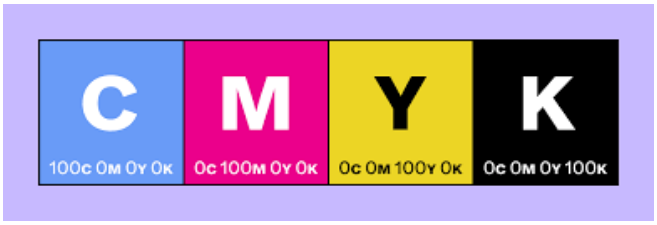



## فضا رنگ CMYK

این فضا رنگ که مخفف cyan, magenta, yellow, key (Black) است، اساس این مدل بر نحوه جذب نور توسط جوهر یا رنگدانه‌ها استوار است. کار آن با یک سطح سفید (مانند کاغذ) آغاز می‌شود و با استفاده از جوهر یا رنگدانه‌ها، طول موج‌های خاصی از نور کم جذب شده و طول موج‌های باقی‌مانده به چشم بیننده بازتاب داده می‌شود.

هر کدام از جوهرهای اشاره شده وظیفه جذب یک نور به خصوص را دارند، مثلا رنگدانه فیروزه‌ای (cyan) وظیفه جذب نور قرمز همراه با بازتاب آبی و سبز است. یا رنگ ارغوانی (magneta) وظیفه جذب رنگ سبز، و آبی و قرمز را بازتاب میدهد. وقتی رنگدانه‌ها روی هم قرار میگیرند، نور بیشتری را جذب میکنند. از نظر تئوری، ترکیب خالص رنگدانه‌های گفته شده باید تمام نور را جذب کرده و رنگ سیاه تولید کند ولی اینکار معمولا امکان پذیر نیست.

هر کدام از رنگدانه‌های گفته شده مقداری ۰ تا ۱۰۰ درصد میتوانند بگیرند و رنگ های جدیدی تولید کنند. از این فضا رنگ معمولا در پرینت گرفتن استفاده میشود. هر چند که در مقایسه با RGB رنگ‌های کمتری را پوشش میدهد، همچنین وقتی RGB را به این فضا رنگ تبدیل میکنیم سبب از دست رفتن درخشندگی بعضی رنگ‌ها میشود و نیاز به تنظیمات مجدد دارد.

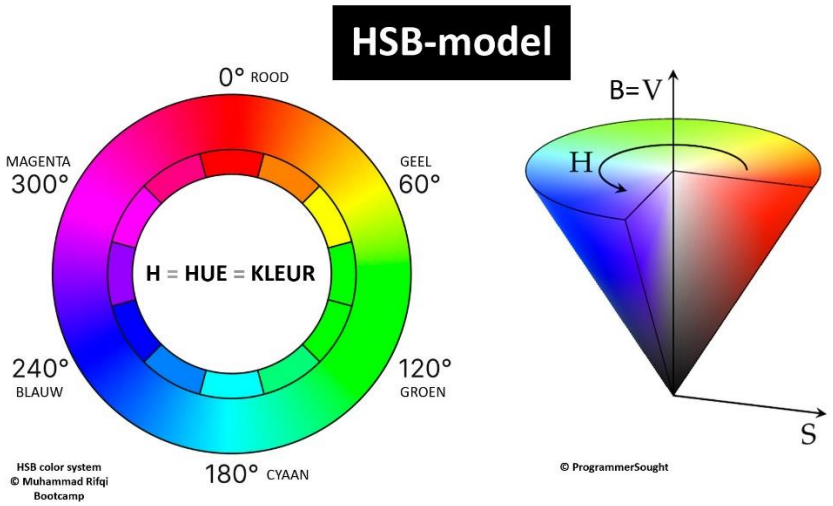



## فضا رنگ HSV

این فضا رنگ که مخفف رنگ (hue)، خالص بودن رنگ یا اشباع (saturation) و ارزش (value) است که البته در بعضی موارد به HSB هم معروف است که به جای مقدار، روشنایی brightness استفاده میشود. این مدل برخلاف RGB یا CMYK است، این مدل تبدیلی از فضای رنگی RGB است که با هدف سازگاری بیشتر با شیوه درک رنگ و ویژگی‌های آن توسط انسان، طراحی شده است. این مدل رنگ‌های را در یک ساختار استوانه‌ای نمایش میدهد.

نحوه کارکرد فضا رنگ به اینگونه است که hue بیانگر خود رنگ خالص است یعنی چیزی که ما بهش رنگ میگیم به نوعی یک زاویه‌ای روی چرخه رنگ تعریف میشود که معمولا در بازه ۰ تا ۳۶۰ درجه قرار دارد. زاویه ۰ یا ۳۶۰ درجه معمولا قرمز و ۱۲۰ درجه سبز و ۲۴۰ درجه آبی است. زوایای میانی نشان دهنده رنگ های میانی هستند مثلا ۶۰ درجه زرد است. مولفه saturation بیانگر شدت یا خلوص رنگ است، این مولفه مشخص میکند که چه مقدار ناخالصی یا خاکستری با خود رنگ خالص ترکیب شده‌اند که معمولا بین ۰ تا ۱۰۰ درصد است. ۰ درصد یعنی رنگ کاملا از اشباغ خارج شده است که دیگر رنگ خالص نیست و یک طیفی به سیاه تا سفید متغیر است، به نوعی انگار صفحه‌ای خاکستری شده است. ۱۰۰ درصد یعنی رنگ در خالص ترین و زنده ترین حالت

ممکن آن رنگ خالص (hue) قرار دارد. مولفه ارزش نشان دهنده روشنی یا تاریکی کلی رنگ است که مقدار آن معمولا بین ۰ تا ۱۰۰ درصد است. ارزش ۰ درصد همیشه سیاه است صرف نظر از مقدار رنگ یا اشباع. ارزش ۱۰۰ درصد نشان دهنده روشن‌ترین حالت ممکن رنگی است که توسط رنگ خالص و اشباع تعریف شده است. کاهش دادن مقدار ارزش، رنگ را تیره‌تر میکند. این فضا رنگ بیشتر مورد استفاده موارد گرافیکی و بسری در کامپیوترها است و برای برنامه نویسی و مباحث پردازش تصویر و بینایی کامپیوتر به کار میرود زیرا میتواند اطلاعات رنگ (hue) را از خالص بودن و میزان روشنایی رنگ جدا میکند و برای تشخیص شی مورد مناسب است. این فضا رنگ برای انسان بسیار شهودی و قابل درک تر است تا بتواند رنگ‌ها را بر اساس درک خودش دستکاری کند، اگر چه از این فضا رنگ نمیتوان برای خروجی فیزیکی مستقیم مانند RGB که در مانیتور ها استفاده میشود یا CMYK که در پرینت گرفتن استفاده میشود. معمولا این فضا رنگ نیاز دارد مقادیر خودش را به RGB تبدیل کند اگر بخواهد که به نمایش در مانیتور در بیاید.




## بخش ب

برای تولید ماسکی که بتواند ناحیه پوست انسان را جدا کند نیاز داریم که با کمک HSV یک محدوده رنگی برای پوست انسان در نظر بگیریم و یک آستانه داشته باشیم تا بتوانیم یک ماسک باینری بسازیم. ماسک باینری از اونجایی نیاز هست که میخواهیم نقاط پوست مثلا سفید باشند و

سایر نقاط که پوست نیستند سیاه باشند. پس در مجموع ۲ رنگ بیشتر نداریم. و بعد به صورت bitwise با کمک عملیات AND میایم و ماسک را اعمال میکنیم، وقتی ماسک اعمال میشود اگر نقطه متناظر در ماسک سفید باشد (یعنی برابر با ۲۵۵) پیکسل مقدار خودش را نگه میدارد ولی اگر نقطه متناظر در ماسک سیاه باشد (یعنی صفر) پیکسل مقدار اولیه خودش را از دست میدهد و آن قسمت سیاه میشود. آستانه هم اگر مقدار داخل آن آستانه باشد به عنوان سفید و اگر نباشد به عنوان سیاه دسته بندی میشود. پوست انسان معمولا محدوده H آن در حدود ۰ تا ۲۵ یا مقادیر مشابه به این خواهد بود. مولفه S نیز مقداری بین ۴۰ تا ۱۵۰ به طور معمول خواهد داشت. مولفه V نیز مقداری بین ۵۰ تا ۲۴۰ خواهد داشت تا از مقادیر کاملا سیاه و کاملا سفید بتواند جلوگیری بکند.

## گزارش کار

در این پروژه ما ابتدا تصویر ورودی را در که در فضا رنگ RGB قرار دارد به فضا رنگ HSV تبدیل میکنیم. زیرا این فضا رنگ برای تشخیص پوست انسان معمولا مناسب تر است زیرا به کمک ۳ مولفه این فضا رنگ میتوانیم پوست انسان را تشخیص دهیم.

Hue بیانگر تن رنگ خواهد بود، تن‌های رنگ پوست انسان، در میان قومیت‌های مختلف، معمولا در محدوده‌ی از رنگ‌ها قرار دارد. saturation بیانگر خلوص یک رنگ است همانطور که قبلا توضیح دادیم. پوست انسان دارای طیف محدودی از اشباع را دارا است که نه به طور کامل خاکستری است (مقدار پایین اشباع) و نه به شدت پر رنگ و شدید (مقدار بالای اشباع). Value نیز میزان روشنایی را مشخص میکند که تحت تاثیر شرایط نوری قرار دارد.

با استفاده از HSV، میشود یک سری آستانه بر اساس hue و saturation در نظر گرفت که کمتر به تغییرات نور پردازی نسبت به RGB حساس است. که همانطور که قبلا توضیح دادیم برای پوست انسان یک محدوده خاصی در نظر گرفتیم و آستانه ها را تعیین کردیم و ماسک را اعمال کردیم.

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
```
✓ 1.8s Python

ابتدا کتابخانه cv2 برای کار با تصویر پیاده‌سازی کردیم. از کتابخانه matplotlib برای گزارش و خروجی تصویری استفاده کردیم.

```python
def segment_skin_hsv(image_path):

    original_image = cv2.imread(image_path)
    if original_image is None:
        print(f"Error: Could not load image from {image_path}")
        return None

    hsv_image = cv2.cvtColor(original_image, cv2.COLOR_BGR2HSV)

    lower_skin = np.array([0, 48, 50], dtype=np.uint8)
    upper_skin = np.array([25, 255, 255], dtype=np.uint8)

    skin_mask = cv2.inRange(hsv_image, lower_skin, upper_skin)

    kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5, 5))
    skin_mask = cv2.morphologyEx(skin_mask, cv2.MORPH_OPEN, kernel)
    skin_mask = cv2.morphologyEx(skin_mask, cv2.MORPH_CLOSE, kernel)


    segmented_skin_image = cv2.bitwise_and(original_image, original_image, mask=skin_mask)

    return original_image, skin_mask, segmented_skin_image
```

بر اساس توضیحات گفته شده، در تابع segment_skin_hsv توضیحات را اعمال کردیم. ابتدا تصویر ورودی را گرفتیم و از RGB به HSV تبدیل کردیم. سپس با کمک lower_skin و upper_skin یک محدوده‌ای برای پوست انسان در نظر گرفتیم (hue: 0-179, s:0-255,v:0-255) البته بر اساس مشاهده سایر خروجی‌های گرفته شده میتوانیم مقداری این بازه‌های را جابجا کنیم. و بعد با استفاده از skin_mask یک ماسک باینری تعریف کردیم. در ادامه با کمک kernel سعی کردیم مقداری عکس را بهتر کنیم و نویز های آن را حذف کنیم در خط اول skin_mask نویزهای کوچک را حذف کردیم و در خط بعدی سعی کردیم حفره‌های کوچک را پر کنیم.

در مرحله بعدی با کمک segmented_skin_image ماسک به عکس اولیه اعمال کردیم. اینکار به صورت bitwise صورت گرفته که نیازمند این است که تصاویر از تعداد کانال یکسان پیروی کنند پس ماسک را به تصویر اولیه RGB اعمال کردیم. در نهایت خروجی را برگرداندیم.

```
result = segment_skin_hsv(image_file)

if result:
    original, mask, segmented = result


    plt.figure(figsize=(15, 5))

    plt.subplot(1, 3, 1)
    plt.imshow(cv2.cvtColor(original, cv2.COLOR_BGR2RGB))
    plt.title('Original Image')
    plt.axis('off')

    plt.subplot(1, 3, 2)
    plt.imshow(mask, cmap='gray')
    plt.title('Skin Mask')
    plt.axis('off')

    plt.subplot(1, 3, 3)
    plt.imshow(cv2.cvtColor(segmented, cv2.COLOR_BGR2RGB))
    plt.title('Segmented Skin')
    plt.axis('off')

    plt.tight_layout()
```

Spaces: 4 Cell 2 of 4 Go Live kite: not installed

با کمک کتابخانه matplotlib و عکس ورودی که عکس لنا است به تابع به عنوان ورودی دادیم و خروجی را گزارش دادیم.

## خروجی

## بخش پ

تغییرات در روشنایی (روشنایی کلی، جهت و دمای رنگ نور) به طور قابل توجهی بر عملکرد الگوریتم HSV ساده که در بخش ب توضیح داده شده است، تأثیر می گذارد. مولفه V مستقیما تحت تاثیر این تغییرات قرار میگیرد. نور کم، همه چیز را تاریک تر می کند و مقادیر V را کاهش می دهد. پیکسل‌هایی که قبلاً در محدوده V بودند (مثلاً ۵۰-۲۴۰) ممکن است به زیر ۵۰ بیایند و باعث شود که ماسک آنها را از دست بدهد. نور تند/ روشن، اشیا را روشن‌تر می‌کند و مقادیر V را افزایش می‌دهد. پیکسل ها ممکن است از آستانه V بالایی (مثلاً ۲۴۰) فراتر روند و دوباره باعث از دست رفتن آنها شود. تابش برجسته (لکه های سفید خالص) قطعا خارج از محدوده V پوست معمولی و همچنین اغلب خارج از محدوده S قرار می گیرند. روشنایی نیز بر اشباع تأثیر می گذارد. نور کم، می تواند اشباع درک رنگ ها را کاهش دهد. پیکسل ها ممکن است زیر آستانه S پایین تر (مثلاً ۴۰) قرار گیرند. حتی مولفه H نیز میتواند تحت تاثیر قرار بگیرد. مثلا اگر نور پردازی رنگی باشد رنگ درک شده پوست را تغییر میدهد و خارج از محدوده H میرود. حتی روشنایی یا تاریکی شدید نیز باعث میشود مولفه H تغییر کند و غیر قابل اعتماد شود. پس تغییرات در روشنایی در درجه اول مقادیر V و S پیکسل‌های پوست را تغییر می‌دهد، به طور بالقوه آنها را به خارج از آستانه‌های ثابت از پیش تعریف‌شده منتقل می‌کند، که منجر به تقسیم‌بندی ضعیف می‌شود (نقاط پوستی از دست رفته یا شامل مناطق غیر پوستی که به‌طور تصادفی به دلیل نور در محدوده قرار می‌گیرند). راه حلی که برای این موضوع میتواند ارائه داد histogram equalizer است که یک تکنیک رایج برای بهبود مقاومت در برابر سطوح مختلف روشنایی و تغییرات نوری است که قبل از اعمال HSV میتوان آن را اعمال کرد که به طور خاص در کانال V تاثیر میگذارد. نحوه یکسان سازی در کانال به این صورت است که مقادیر روشنایی را دوباره توزیع می‌کند تا کل محدوده ۰-۲۵۵ را به طور یکنواخت‌تر پوشش دهد، کنتراست را افزایش می‌دهد و حساسیت تصویر را نسبت به روشنایی کلی کم یا زیاد کمتر می‌کند. که در قسمت گزارش کار پیاده سازی عملی آن را خواهیم دید:

# گزارش کار

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

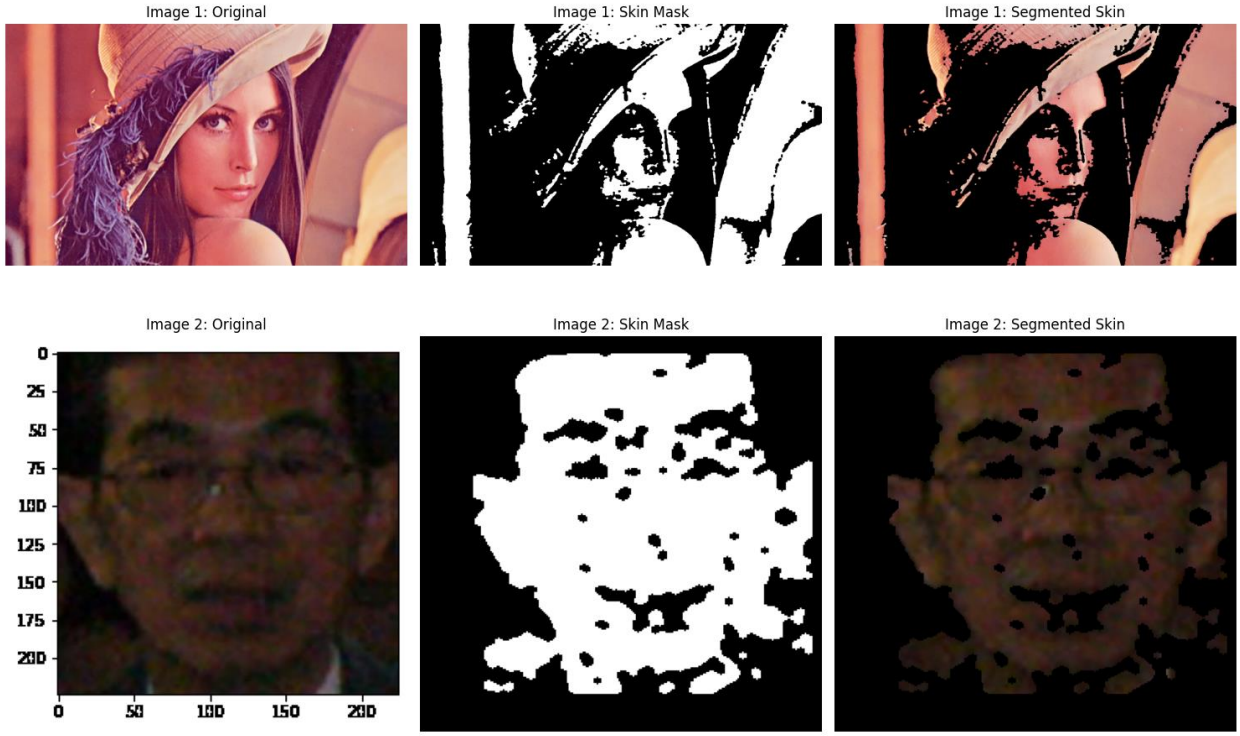
همانند بالا کتابخانه های مورد نظر را وارد کردیم.

```python
def segment_skin_hsv_robust(image_path):

    original_image = cv2.imread(image_path)
    if original_image is None:
        print(f"Error: Could not load image from {image_path}")
        return None

    hsv_image = cv2.cvtColor(original_image, cv2.COLOR_BGR2HSV)

    h, s, v = cv2.split(hsv_image)

    v_equalized = cv2.equalizeHist(v)

    hsv_equalized = cv2.merge([h, s, v_equalized])

    lower_skin = np.array([0, 48, 50], dtype=np.uint8)
    upper_skin = np.array([25, 255, 255], dtype=np.uint8)

    skin_mask = cv2.inRange(hsv_equalized, lower_skin, upper_skin)

    kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5, 5))
    skin_mask = cv2.morphologyEx(skin_mask, cv2.MORPH_OPEN, kernel)
    skin_mask = cv2.morphologyEx(skin_mask, cv2.MORPH_CLOSE, kernel)

    segmented_skin_image = cv2.bitwise_and(original_image, original_image, mask=skin_mask)


    return original_image, skin_mask, segmented_skin_image
```

عکس را بارگذاری کردیم. در ادامه از RGB به HSV تبدیل کردیم. بعد HSV را به ۳ کانال جدا کردیم. در مرحله بعد HE یا همان histogram equalization که روش پیشنهادی برای بهبود مقاوم سازی تصویر ما نسبت به نور بود را اعمال کردیم بر روی کانال v. بعد با کمک hsv_equalizied کانال ها را دوباره یکی کردیم.

در ادامه یک محدوده ای برای رنگ پوست انسان با کمک lower_skin و upper_skin تعریف کردیم. با کمک skin_mask یک ماسک باینری v-eqalized شده درست کردیم. در نهایت ماسک را به تصویر اصلی که RGB بود اعمال کردیم و خروجی بهبود یافته شده که نسبت به تغییرات نوری مقاوم تر هست در خروجی قابل مشاهده است.

## خروجی




در عکس هایی که میزان روشنایی آنها متفاوت است بهتر میتوان تاثیر روش پیشنهاد را مشاهده کرد.