Assignment 5

Ratchanon Panmas 6434480323

1. Write a program to convert the image “fruit.jpg” to its complementary colors and display it on the screen.

การแก้ไขสีของภาพให้กลายเป็น Complementary Color นั้นทำได้หลายแบบจากเนื้อหาที่ผ่านมาขอนำเสนอการสร้างรูปภาพ Complementary 2 แบบภายใต้ RGB Color Space และ HSI Color Space

แบบที่ 1 ภายใต้ RGB Color Space

```
def complementary_color_RGB(img):
    # Split the image into three channels
    b, g, r = cv.split(img)

    b = cv.bitwise_not(b)
    g = cv.bitwise_not(g)
    r = cv.bitwise_not(r)

    # Merge the three channels
    img = cv.merge([b, g, r])

    return img
```

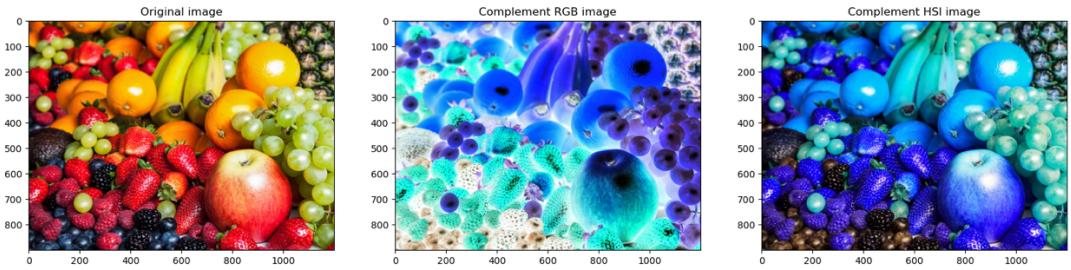
ในการเปลี่ยนสีเป็น Complementary Color ภายใต้ RGB Color Space นั้นสามารถทำได้โดยนำค่าสูงสุดของแต่ละ Channel มาลบกับค่าดั้งเดิมของ Channel นั้นๆ ผลลัพธ์ที่ได้เมื่อนำทั้ง 3 Channel มารวมกันก็จะสีที่เป็น Complementary ของรูปเดิม

แบบที่ 2 ภายใต้ HSI Color Space

```
def complementary_color_HSI(img):
    complementary_img = img.copy()

    h, s, i = complementary_img[:, :, 0], complementary_img[:, :, 1],
complementary_img[:, :, 2]
    h = (h + 0.5) % 1.0

    return complementary_img
```

ในการเปลี่ยนสีเป็น Complementary Color ภายใต้ HSI Color Space นั้นจะเป็นการหา Complementary Hue เนื่องจาก Hue เป็น Channel ที่กำหนดสีของ pixel นั้นๆ

ผลลัพธ์ที่ได้จากการทำ Complementary Color

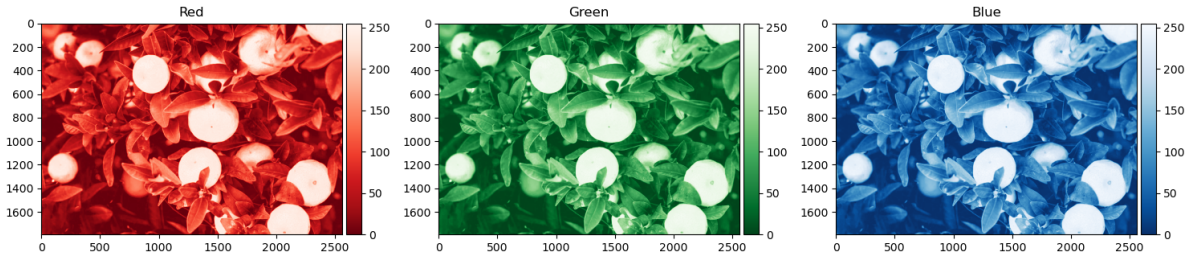


จากภาพจะเห็นได้ว่าผลลัพธ์ที่ได้จากการ Complementary ต่าง Color Space กันจะได้ผลลัพธ์ที่ต่างกัน ในส่วนของ Color Space ที่เป็น RGB นั้นผลลัพธ์ที่ได้จะเป็นสีตรงข้ามแบบ Negative Value บริเวณที่ขาวหรือสว่างก็จะกลายเป็นสีดำ แต่ใน HSI Color Space นั้นจะเป็นการกลับแค่ค่า Hue ทำให้ส่วนที่สว่างก็ยังสว่างอยู่เนื่องจากไม่ได้มีการเปลี่ยนแปลงค่า Saturation และ Intensity

2. Use color slicing to segment the oranges from the image “oranges.jpg” by keeping the colors of the oranges to be same but changing the colors of other fruits to blue color (0,0,0.5). Find the most suitable range of orange color using your judgement.

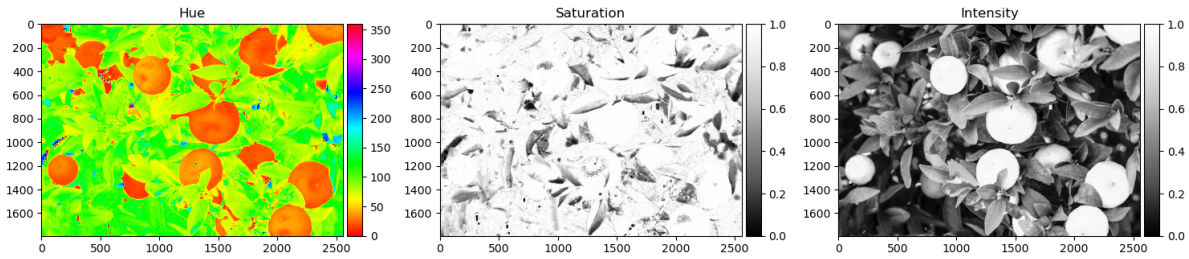
ภาพส้มที่เรามีเฉดสีของผลส้มนั้นมีความแตกต่างกันค่อนข้างมาก เนื่องจากผลส้มนั้นอยู่ทั้งในบริเวณที่สว่างและมืด เลยลองแยก Channel ของภาพใน Color space ที่แตกต่างกัน

ใน RGB Color Space



สังเกตบริเวณที่เป็นผลส้มจะเห็นว่ามีค่าทั้งใน R G B ค่อนข้างสูง ใน Red Channel นั้นจะพบว่ามีเฉพาะส่วนที่เป็นส้มเท่านั้นที่มีค่าสีแดงค่อนข้างสูง ในส่วนอื่นๆนั้นจะมีสีแดงน้อยกว่า แต่ใน Blue Channel นั้น Range ของค่อนข้างกว้าง จากการวิเคราะห์ RGB แต่ละ Channel จะพบว่าเราสามารถ slice ส่วนที่ไม่ใช่ส้มออกไปได้ส่วนมากโดยกำหนดค่า Red ให้สูงและมี Range ที่ค่อนข้างแคบได้

ใน HSI Color Space



เนื่องจากลองนำรูป orange.jpg ที่มีนำมาแยกออกเป็น RGB Channel พบว่าแยกส่วนที่เป็นส้มนั้นได้ค่อนข้างยาก เลยลองนำมาแยกใน HSI Color Space จะเห็นได้ว่าค่า Hue ส่วนที่เป็นสีส้มนั้นค่อนข้างแคบและหากสังเกตบริเวนที่เป็นสีส้มแล้วจะเห็นได้ว่าจะเป็นส่วนที่เป็นผลส้มทั้งหมดไม่ว่างส้มนั้นจะอยู่ในเงาหรือไม่ได้อยู่ในเงาก็ตาม

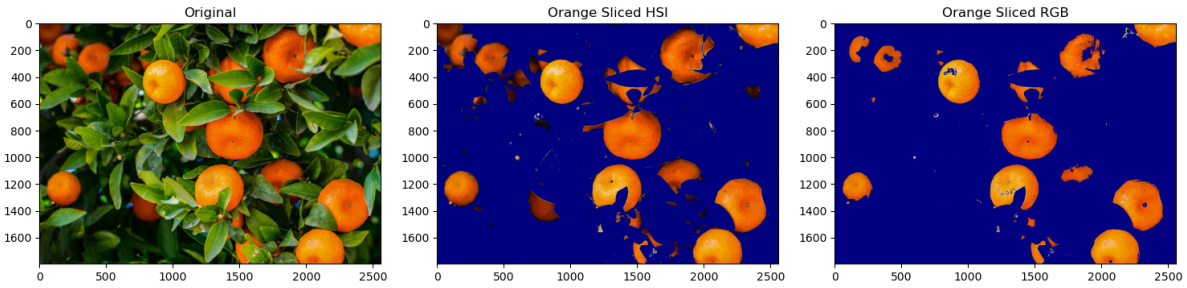
แบบที่ 1 Color Slicing โดยใช้ HSI Color Space โดยการ Set Threshold ของ Hue ในช่วงที่ต้องการ

```
def filterOrange(img, hue=[0, 50]):

    min_hue, max_hue = hue

    hue_range = [min_hue / 360, max_hue/360]
    saturation_range = [0, 1]
    intensity_range = [0, 1]

    width, height, channel = img.shape

    # Create a new image
    new_img = np.zeros((width, height, channel), dtype=np.float64)

    for i in range(width):
        for j in range(height):
            # Get RGB value
            hue, saturation, intensity = img[i, j, 0], img[i, j, 1], img[i, j, 2]

            # Filter out not orange
            if hue_range[0] <= hue <= hue_range[1] and saturation_range[0] <=
saturation <= saturation_range[1] and intensity_range[0] <= intensity <=
intensity_range[1]:
                new_img[i, j] = [hue, saturation, intensity]
            else:
                new_img[i, j] = [240/360, 1, 0.5]

    return new_img
```

แบบที่ 2 Color Slicing โดยใช้ RGB Color Space โดยการ Set ค่าของ RGB ของสีที่ต้องการและมีการ Set Range ระยะของสีที่สามารถเพี้ยนไปได้จากค่าที่ต้องการโดยมีการ Set ให้ค่า Range ของสีในแต่ละ Channel นั้นมีระยะไม่เท่ากัน

```
def colorSlicing(img, color, colorRange):

    r_range, g_range, b_range = colorRange
    r_color, g_color, b_color = color

    width, height, channel = img.shape

    # Create a new image
    new_img = np.zeros((width, height, channel), dtype=np.uint8)

    for i in range(width):
        for j in range(height):
            r, g, b = img[i, j]

            diff_r = abs(r - r_color)
            diff_g = abs(g - g_color)
            diff_b = abs(b - b_color)

            if diff_r <= r_range and diff_g <= g_range and diff_b <= b_range:
                new_img[i, j] = [r, g, b]
            else:
                new_img[i, j] = [0, 0, 128]

    return new_img
```

ผลลัพธ์จากการทำ Color Slicing



เปรียบเทียบผลลัพธ์การทำ Color Slicing จะเห็นได้อย่างชัดเจนว่าใน HSI Color Space นั้นจะสามารถเก็บส่วนที่เป็นผลส้มได้ครบกว่าและภาพที่ได้จะมีความคมชัดกว่า ไม่มีส่วนที่ไม่ใช่ส้มเข้ามาในภาพ และผลส้มที่ได้จะไม่แหว่งมีรูปทรงที่สมบูรณ์กว่า RGB

ค่าที่ใช้ในการทำ Slicing ใน RGB Color Space

ค่าสีส้มที่ใช้ คือ (225, 100, 0)
Range ของสีที่ใช้ คือ (30, 100, 255)

ค่าที่ใช้ในการทำ Slicing ใน HSI Color Space

Hue จะอยู่ในช่วง [5, 50]

Appendix

Code ฉบับเต็มที่ใช้ในการทำงาน

Code ในการทำ Complementary Color
https://github.com/ratchanonp/imageprocessing/blob/main/Assignment_5/Assignment_5_1.ipynb

Code ในการทำ Color Slicing
https://github.com/ratchanonp/imageprocessing/blob/main/Assignment_5/Assignment_5_2.ipynb

ผลลัพธ์ที่ได้

Directory ของผลลัพธ์ที่ได้
https://github.com/ratchanonp/imageprocessing/tree/main/Assignment_5/output