Assignment 02

ต่อไปนี้จะเป็นการแสดงในส่วนของโปรแกรมในเรื่องของ Sampling & Quantization ด้วยภาษา python

Libraries

- Scikit-image: ใช้ในการรับภาพเป็นข้อมูลนำเข้าและปรับขนาดภาพในส่วนของการทำ Sampling
- Scikit-learn: ใช้ฟังก์ชัน KMean เพื่อในการแบ่งสีเป็นกลุ่มๆเพื่อเกลี่ยสีในการทำ Quantization
- Matplotlib เพื่อการแสดงผลทางหน้าจอของโปรแกรม

Program

ในส่วนของโปรแกรมจะแบ่งการทำงานออกเป็นสองส่วนโดยแบ่งเป็นส่วนของ Sampling และส่วนของ Quantization โดยแบ่งเป็นส่วนละฟังก์ชันและสองฟังก์ชันนี้ไม่เกี่ยวข้องกันโดยสิ้นเชิง ในเรื่องของการเรียกใช้งานโปรแกรมนั้นจะเป็นการเรียกชื่อฟังก์ชันเท่านั้นภายใต้คำสั่ง if __name___ == '__main__': ดังรูป

```
if __name__ == '__main__':
    sampling()
```

รูปที่ 1 การเรียกใช้งานฟังก์ชัน

```
def sampling():
    img = io.imread("MrGenie.JPG")
    fig, axes = plt.subplots(nrows=2, ncols=3)
    ax = axes.ravel()
    ax[0].imshow(img)
    ax[0].set_title("Original")

    for k in range(5, 0, -1):
        print(k)
        img_resized = resize(img, (img.shape[0] // 2**k, img.shape[1] // 2**k),
                             anti_aliasing=True)

        ax[k].imshow(img_resized)
        ax[k].set_title("x"+ str(2**k))

    plt.tight_layout()
    plt.show()
```

รูปที่ 2 ฟังก์ชัน sampling()

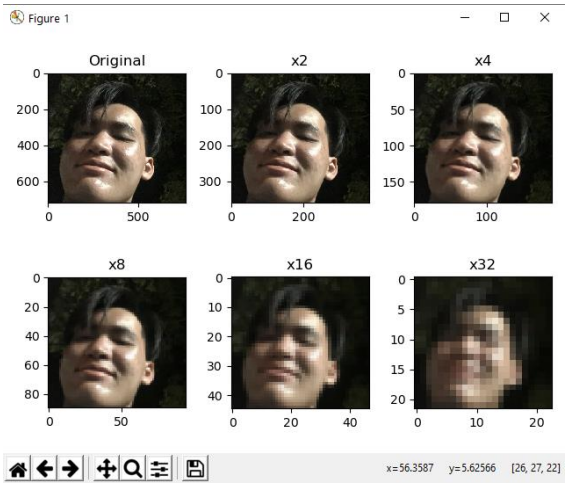
- ฟังก์ชัน sampling() จะทำงานโดยการเริ่มรับภาพต้นจากskimage.io.imread จากนั้นประมวลผลภาพโดยใช้ฟังก์ชัน resize จากโมดูล transform ของ scikit-image โดยลดขนาดจากความกว้างและความยาวโดยคิดเป็นด้านๆ จากนั้นใช้ for loop เพื่อให้ได้ภาพหลายภาพเพื่อให้เห็นถึงความแตกต่างของการลดขนาดแต่ละสัดส่วน ในส่วนของการแสดงผลนั้นใช้ pyplot จาก matplotlibโดยใช้ฟังก์ชัน subplots เพื่อให้แสดงผลได้หลายภาพใน figure เดียวกัน

```
def quantization():
    original = io.imread("MrGenie.JPG")
    fig, axes = plt.subplots(nrows=2, ncols=3)
    ax = axes.ravel()
    ax[5].imshow(original)
    ax[5].set_title("original")

    for k in range(1, 6, 1):
        n_colors = 2**k
        arr = original.reshape((-1, 3))
        kmeans = KMeans(n_clusters=n_colors, random_state=42).fit(arr)
        labels = kmeans.labels_
        centers = kmeans.cluster_centers_
        less_colors = centers[labels].reshape(original.shape).astype('uint8')

        print(k-1)
        ax[k-1].imshow(less_colors)
        ax[k-1].set_title(str(n_colors) + " colors")

    plt.tight_layout()
    plt.show()
```

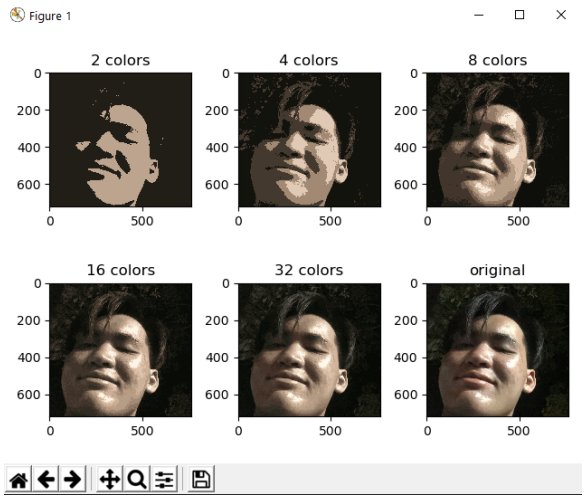
รูปที่ 3 ฟังก์ชัน quantization()

- ฟังก์ชัน quantization() เป็นฟังก์ชันที่เริ่มทำงานคล้ายกับฟังก์ชัน sampling() ในส่วนของการรับข้อมูลจากนั้นในส่วนของการประมวลผลภาพจะใช้ฟังก์ชัน KMeans จาก scikit-learn โมดูล cluster เพื่อแบ่งกลุ่มของสีให้เล็กลงจากจำนวนสีทั้งหมด

ผลลัพธ์



รูปที่ 5 ผลลัพธ์การ Sampling



รูปที่ 6 ผลลัพธ์การ Quantization

อ้างอิง

https://scikit-image.org/docs/stable/auto_examples/transform/plot_rescale.html

from Tonechas

https://stackoverflow.com/questions/48222977/python-converting-an-image-to-use-less-colors

https://scikit-learn.org/stable/auto_examples/cluster/plot_color_quantization.html