



KHOA CÔNG NGHỆ THÔNG TIN

# BÁO CÁO ĐỒ ÁN 1: COLOR COMPRESSION TOÁN ỨNG DỤNG VÀ THỐNG KÊ

GV hướng dẫn: Thầy Vũ Quốc Hoàng

Thầy Nguyễn Văn Quang Huy

Thầy Lê Thanh Tùng

Cô Phan Thị Phương Uyên



Thông tin của sinh viên:

Huỳnh Tấn Vinh – 2012766

Lớp: 20CLC08

## Mục Lục

I. Thông tin tổng quát: .....	2
II. Ý tưởng thực hiện, mô tả các hàm.....	2
1. Ý tưởng thực hiện. ....	2
2. Mô tả các hàm.....	2
a) Import các thư viện cần thiết.....	2
b) Đọc và hiển thị hình ảnh. ....	3
c) Hàm so sánh hai mảng: .....	3
d) Hàm tính toán centroids mới:.....	4
e) Thuật toán K-Means: .....	4
f) Hàm lưu ảnh. ....	5
g) Hàm main .....	5
III. Kết quả và minh họa .....	6
IV. Tài liệu tham khảo. ....	8

## I. Thông tin tổng quát:

Đề án 1 – Color Compression.

Giới thiệu: Một bức ảnh có thể lưu trữ dưới ma trận của các điểm ảnh. Có nhiều loại điểm ảnh được sử dụng trong thực tế, ví dụ: ảnh xám, ảnh màu, ...

Yêu cầu: Cài đặt chương trình giảm số lượng màu cho ảnh sử dụng thuật toán K-Means.

Môi trường thực hiện: Sử dụng Jupyter Notebook. Python 3.9.12.

Đánh giá mức độ hoàn thành: 100%.

Tổng số trang báo cáo: 10 trang (7 trang nội dung chính).

## II. Ý tưởng thực hiện, mô tả các hàm.

### 1. Ý tưởng thực hiện.

- Sử dụng thư viện **PIL** để xử lý hình ảnh.
- Sử dụng thư viện **numpy** để xử lý tính toán.
- Reshape ảnh từ một scalar ba chiều thành một mảng 2 chiều.
- Sử dụng 2 cách khởi tạo centroids.
  - Random màu được lấy có sẵn trong ảnh
  - Random màu được lấy trong khoảng [0,255].
- Thực hiện init cho centroids là k điểm màu. Với k là số cluster (3, 5 hoặc 7).
- Thực hiện vòng lặp với max\_iter lần. Đối với mỗi điểm ảnh tính được sau mỗi lần lặp, tính distance và lấy mean của các cụm ảnh gán cho label mới. Đồng thời cập nhật lại centroids.

### 2. Mô tả các hàm.

a) Import các thư viện cần thiết.

```
## Import thư viện
import matplotlib.pyplot as plt
import numpy as np
from PIL import Image
```

- **matplotlib.pyplot**: là một thư viện vẽ đồ thị cho những người làm việc với Python và NumPy.
- **numpy**: Cho phép làm việc hiệu quả với ma trận và mảng, đặc biệt là dữ liệu ma trận và mảng lớn với tốc độ xử lý nhanh hơn nhiều lần khi chỉ sử dụng “core Python” đơn thuần.
- **PIL**: thư viện cho phép xử lý hình ảnh bằng Python.

b) Đọc và hiển thị hình ảnh.

```
## Đọc ảnh và hiển thị ảnh
file_name = input("Nhập tên file ảnh muốn mở: ") # Có 2 file ảnh gốc là: "Landscape.jpg" và "picture.jpg".
raw_img = Image.open('./' + file_name)
plt.imshow(raw_img)
img = np.array(raw_img)
#Lấy chiều cao, chiều dài của ảnh
heightImg, widthImg = img.shape[0], img.shape[1]
#Reshape lại hình ảnh từ một scalar 3 chiều thành mảng 2 chiều
img = img.reshape( heightImg*widthImg, img.shape[2])
```

- Người dùng nhập tên file của ảnh muốn mở. Trong bài này có 2 hình ảnh gốc là “landscape.jpg” và “picture.jpg”.

Hình ảnh minh họa được hiển thị (landscape.jpg):



c) Hàm so sánh hai mảng:

```
#So sánh 2 mảng
def checkArrayEqual(arr1, arr2):
    return np.array_equiv(arr1, arr2)
```

Dùng để kiểm tra 2 centroid trước và sau có bằng nhau hay không? Nếu bằng nhau thì dừng lại, không thực hiện tính toán centroid mới. Điều này giúp tiết kiệm được chi phí. Thuật toán sẽ không cần phải thực hiện số lần lặp tối đa max\_iter.

d) Hàm tính toán centroids mới:

```
#Tìm bộ centroids mới
def findNewCentroids(max_iter, img_1d, centroids, k_clusters):

    for i in range(max_iter):
        d = np.linalg.norm(img_1d - centroids[:, np.newaxis], axis = 2)
        labels = np.argmin(d, axis = 0)

        #Cập nhật giá trị trung bình
        averages = []
        for k in range(k_clusters):
            averages.append(img_1d[labels == k].mean(axis = 0))
        averages = np.array(averages)
        old_centroid = centroids
        for i in range(k_clusters):
            if len(averages[i]) != 0:
                centroids[i] = averages[i]

        if(checkArrayEqual(old_centroid, centroids)):
            break
    return labels
```

- Sử dụng vòng lặp với max\_iter lần lặp. Với mỗi lần lặp thực hiện tính toán và cập nhật giá trị trung bình. Nếu len(averages[i]) khác 0 (có sự thay đổi), thực hiện gán centroids[i] bằng với giá trị trung bình vừa mới tính được.
- Sau đó, thực hiện kiểm tra nếu centroid trước đó (old\_centroid) bằng với centroids mới tính được thì break. Thoát khỏi vòng lặp. Điều này giúp tiết kiệm được chi phí thay vì thực hiện max\_iter lần lặp, chương trình có thể thực hiện với ít số lần lặp hơn.

e) Thuật toán K-Means:

```
def kmeans(img_1d, k_clusters, max_iter, init_centroids='random'):

    # Khởi tạo k cluster từ trên ảnh
    if init_centroids == 'in_pixels':
        centroids = img_1d[np.random.choice(img_1d.shape[0], size = k_clusters, replace = False)]
    # Khởi tạo k cluster random
    elif init_centroids == 'random':
        centroids = np.random.randint(0,255,size=(k_clusters,img_1d.shape[1]))

    #Tìm bộ centroids mới
    labels = findNewCentroids(max_iter, img_1d, centroids, k_clusters)
    return centroids, labels
```

Trong thuật toán K-Means mỗi cụm dữ liệu được đặt trưng bởi một tâm (centroid). Tâm là điểm đại diện duy nhất cho một cụm và có giá trị bằng trung bình của toàn bộ các quan sát nằm trong cụm.

- Khởi tạo k tâm cluster từ trên ảnh. Sử dụng 2 cách:
  - in\_pixels: random màu được lấy trong ảnh.
  - radom: random màu được lấy trong khoảng [0,255].
- Tìm bộ centroids mới.

f) Hàm lưu ảnh.

```
def saveImage(test_img):
    img_type = input("Nhập định dạng file ảnh muốn lưu: ") # định dạng "pdf" hoặc định dạng "jpg, png".
    if (img_type == "pdf"):
        save_img = Image.fromarray(test_img)
        file_name = input("Nhập tên file ảnh muốn lưu: ")
        save_img = save_img.convert('RGB')
        save_img.save('./' + file_name + '.pdf')
        print("Ảnh đã được lưu!")

    elif (img_type == "jpg"):
        save_img = Image.fromarray(test_img)
        file_name = input("Nhập tên file ảnh muốn lưu: ")
        save_img.save('./' + file_name + '.jpg')
        print("Ảnh đã được lưu!")

    elif (img_type == "png"):
        save_img = Image.fromarray(test_img)
        file_name = input("Nhập tên file ảnh muốn lưu: ")
        save_img.save('./' + file_name + '.png')
        print("Ảnh đã được lưu!")

    else:
        print("Định dạng không chính xác!")
```

- Cho phép người nhập định dạng của file ảnh muốn lưu.
- Có 3 định dạng được phép lưu: pdf, jpg và png.
- Nếu nhập sai ba định dạng trên. Chương trình sẽ thông báo định dạng không chính xác.

g) Hàm main

```
def main():
    k_cluster = int(input("Nhập số màu ảnh: "))
    test_img = img.copy()
    init_centroids = input("Nhập cách khởi tạo centroid: ") #Hai cách khởi tạo: in_pixels hoặc random

    centroids, labels = kmeans(test_img, k_cluster, 10, init_centroids)

    ## Gắn Lại Label cho ảnh
    for k in range(centroids.shape[0]):
        test_img[labels == k] = centroids[k]

    test_img = test_img.astype("uint8")
    test_img = test_img.reshape(heightImg, widthImg, 3)
    plt.imshow(test_img)
    saveImage(test_img)
```

- Cho phép người dùng nhập số màu ảnh và các khởi tạo centroids.
- Thực hiện gắn lại label cho ảnh.
- Show ảnh sau khi giảm số lượng màu.
- Thực hiện các thao tác lưu file ảnh.

### III. Kết quả và minh họa

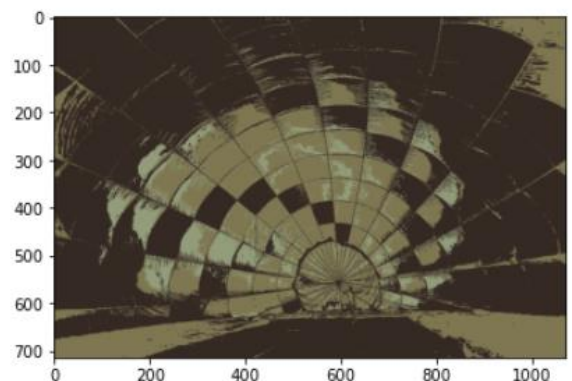
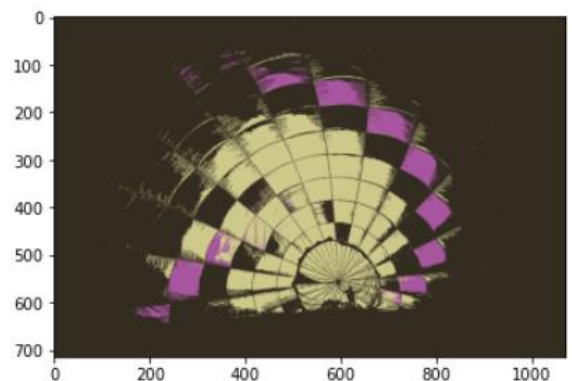
- Nhập file ảnh để mở:

Nhập tên file ảnh muốn mở: picture.jpg

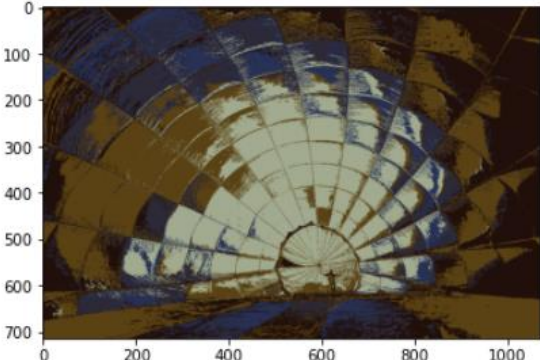

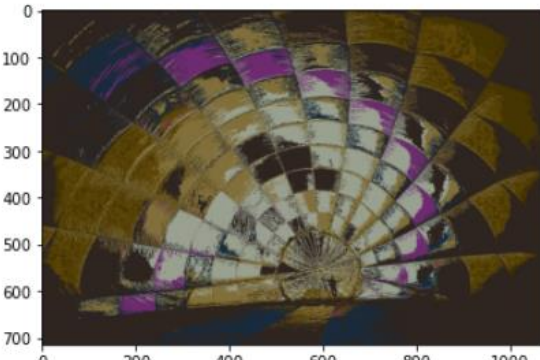



(Hình ảnh gốc)

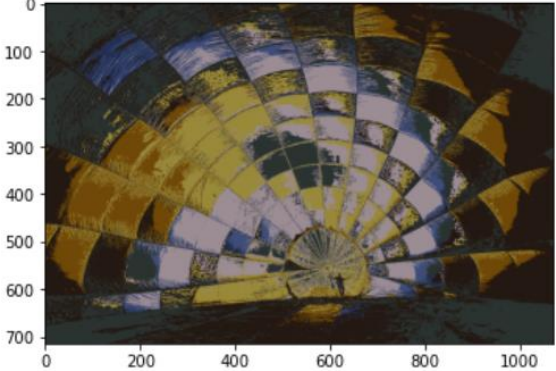
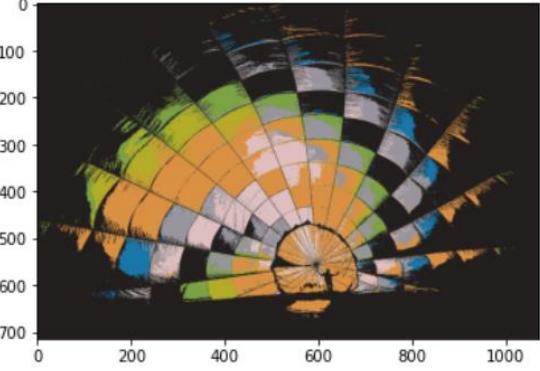
- ❖ Một số hình ảnh minh họa cho các trường hợp  $k = 3, 5, 7$ :

Cách khởi tạo centroid: in_pixels	Cách khởi tạo centroid: random
<pre>main() Nhập số màu ảnh: 3 Nhập cách khởi tạo centroid: in_pixels Nhập định dạng file ảnh muốn lưu: pdf Nhập tên file ảnh muốn lưu: picture3 Ảnh đã được lưu!</pre> 	<pre>main() Nhập số màu ảnh: 3 Nhập cách khởi tạo centroid: random Nhập định dạng file ảnh muốn lưu: jpg Nhập tên file ảnh muốn lưu: picture_jpg_3 Ảnh đã được lưu!</pre> 
K = 3, định dạng lưu file ảnh: pdf	K = 3, định dạng lưu file ảnh: jpg








<pre>main()</pre> <p>Nhập số màu ảnh: 5          Nhập cách khởi tạo centroid: in_pixels          Nhập định dạng file ảnh muốn lưu: png          Nhập tên file ảnh muốn lưu: picture5a          Ảnh đã được lưu!</p> 	<pre>main()</pre> <p>Nhập số màu ảnh: 5          Nhập cách khởi tạo centroid: random          Nhập định dạng file ảnh muốn lưu: jpg          Nhập tên file ảnh muốn lưu: picture5b          Ảnh đã được lưu!</p> 
K= 5, định dạng lưu file ảnh: png	K= 5, định dạng lưu file ảnh: jpg
<pre>main()</pre> <p>Nhập số màu ảnh: 7          Nhập cách khởi tạo centroid: in_pixels          Nhập định dạng file ảnh muốn lưu: pdf          Nhập tên file ảnh muốn lưu: picture7a          Ảnh đã được lưu!</p> 	<pre>main()</pre> <p>Nhập số màu ảnh: 7          Nhập cách khởi tạo centroid: random          Nhập định dạng file ảnh muốn lưu: png          Nhập tên file ảnh muốn lưu: picture7b          Ảnh đã được lưu!</p> 
K = 7, định dạng lưu file ảnh: pdf	K = 7, định dạng lưu file ảnh: png



<pre>main() Nhập số màu ảnh: 7 Nhập cách khởi tạo centroid: in_pixels Nhập định dạng file ảnh muốn lưu: cd Định dạng không chính xác!</pre>  <p>K = 7, định dạng lưu file ảnh không chính xác</p>	<pre>main() Nhập số màu ảnh: 7 Nhập cách khởi tạo centroid: random Nhập định dạng file ảnh muốn lưu: sd Định dạng không chính xác!</pre>  <p>K= 7, định dạng lưu file ảnh không chính xác</p>
--	---

❖ Các file ảnh đã được lưu từ những minh họa trên:

<input type="checkbox"/>	 <a href="#">picture3.pdf</a>
<input type="checkbox"/>	 <a href="#">picture5a.png</a>
<input type="checkbox"/>	 <a href="#">picture5b.jpg</a>
<input type="checkbox"/>	 <a href="#">picture7a.pdf</a>
<input type="checkbox"/>	 <a href="#">picture7b.png</a>
<input type="checkbox"/>	 <a href="#">picture_jpg_3.jpg</a>

❖ Nhận xét các kết quả trên: Hình ảnh đã được giảm số lượng màu. Số lượng màu trong hình ảnh chính xác, đúng với yêu cầu.

#### IV. Tài liệu tham khảo.

1. [https://github.com/boosuro/image\\_color\\_compression\\_kmeans/blob/master/image\\_color\\_compression\\_kmeans.ipynb](https://github.com/boosuro/image_color_compression_kmeans/blob/master/image_color_compression_kmeans.ipynb)
2. <https://github.com/samstiv/Color-Compression-Image/tree/master/Compress%20Image>
3. [https://github.com/Yadukrishnan1/Color-Compression-Unsupervised-ML/blob/master/image\\_compress.ipynb](https://github.com/Yadukrishnan1/Color-Compression-Unsupervised-ML/blob/master/image_compress.ipynb)
4. <https://github.com/t3bol90/ST-MA-Lab02>
5. <https://github.com/kieuconghau/color-compression>

6. [https://phamdinhkhanh.github.io/deepai-book/ch\\_ml/KMeans.html](https://phamdinhkhanh.github.io/deepai-book/ch_ml/KMeans.html)
7. <https://www.youtube.com/watch?v=blioxQE1ow4>
8. <https://datatofish.com/images-to-pdf-python/>
9. <https://www.delftstack.com/howto/numpy/save-numpy-array-as-image/>