KHOA CÔNG NGHỆ THÔNG TIN

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN – ĐHQG TPHCM



**TOÁN ỨNG DỤNG VÀ THỐNG KÊ CHO   
CÔNG NGHỆ THÔNG TIN**  
  
**ĐỒ ÁN 1: COLOR COMPRESSION**

**LỚP: 21CLC08**

**HỌ VÀ TÊN: NGÔ QUỐC QUÝ**

**MSSV: 21127679**

**Mục lục**

[**I.** **Ý tưởng thực hiện, mô tả các hàm:** 3](#_Toc140435473)

[**a.** **Ý tưởng thực hiện:** 3](#_Toc140435474)

[**b.** **Mô tả các hàm:** 3](#_Toc140435475)

[**II.** **Kết quả:** 5](#_Toc140435476)

[**a.** **Ảnh gốc:** 5](#_Toc140435477)

[**b.** **Ảnh sau khi được nén với k = 3:** 5](#_Toc140435478)

[**c.** **Ảnh sau khi được nén với k = 5:** 6](#_Toc140435479)

[**d.** **Ảnh sau khi được nén với k = 7:** 6](#_Toc140435480)

[**III.** **Nhận xét:** 6](#_Toc140435481)

[**IV.** **Tài liệu tham khảo:** 6](#_Toc140435482)

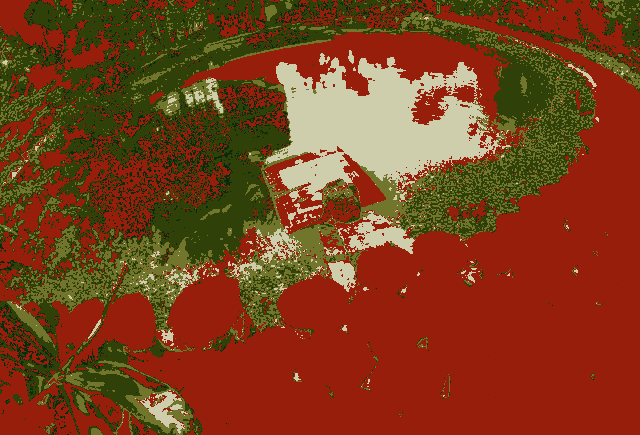
1. **Ý tưởng thực hiện, mô tả các hàm:**
   1. **Ý tưởng thực hiện:**

* Được xây dựng dựa trên thuật toán Kmeans Clustering, phân cụm khi chúng ta không biết labels của từng điểm dữ liệu
* Mô tả thuật toán:
  + Khởi tạo các centroids ngẫu nhiên
  + Chạy vòng lặp đến max\_iterator
    - Gán nhãn cho từng điểm dữ liệu
    - Cập nhật dữ liệu của các centroids.
    - Kiểm tra điểm hội tụ (tránh lãng phí bộ nhớ, thời gian)
  1. **Mô tả các hàm:**
* **Hàm main**:
  + Gọi hàm **input\_image**, thực hiện nhập hình ảnh.
  + Gọi hàm **handle\_and\_compress\_image** để xử lí hình ảnh được nhập vào.
* **Hàm input\_image:**
  + Cho phép người dùng nhập tên tập tin hình ảnh (phải cùng 1 đường dẫn với file ipynb).
  + Thực hiện mở hình ảnh, nếu không có thì báo lỗi.
* **Hàm handle\_and\_compress\_image:**
  + Đầu vào:
    - image: hình ảnh có được sau khi gọi hàm **input\_image**
  + Đầu ra:
    - Không có giá trị trả về
  + Hàm thực hiện chuyển đổi hình ảnh sang dạng ma trận, gọi hàm kmeans lấy về hai giá trị centroids và labels, sau đó sử dụng 2 giá trị trên để phục hồi về dạng ma trận 3 chiều (row, col, rgb(3)).
  + Tiếp theo, yêu cầu người dùng nhập định dạng muốn lưu trữ hình ảnh sau khi nén (png, pdf).
  + Sau khi lưu trữ, tiến hành in ra console.
* **Hàm kmeans:**
  + Đầu vào:
    - img\_1d: tập dữ liệu ban đầu.
    - k\_cluster: số lượng màu muốn nén.
    - max\_iter: số vòng lặp tối đa khi phân cụm.
    - init\_centroids: cách khởi tạo các center (truyền vào 2 giá trị random và in\_pixels).
  + Đầu ra:
    - centroids: các trung tâm sau khi nén.
    - labels: các nhãn dán cho điểm dữ liệu trung tâm.
  + Hàm này đầu tiên sẽ khởi tạo một mảng centroids bất kì (được lấy giá trị random theo từng đầu vào (random hoặc in\_pixels).
  + Thực hiện vòng lặp, tìm có khoảng cách gần nhất để gán labels và thực hiện tính toán lại các cụm centroids.
* **Hàm compute\_distance:**
  + Đầu vào:
    - img\_1d: tập dữ liệu ban đầu
    - centroids: các cụm trung tâm
    - index: vị trí của phần tử trong tập dữ liệu ban đầu
    - cluster: vị trí của phần tử trong các cụm trung tâm
  + Đầu ra:
    - Khoảng cách của các điểm khác với điểm trung tâm.
  + Thực hiện tính toán khoảng cách .
  + Sử dụng cách tính manhattan distance.
* **Hàm assign\_labels:**
  + **Đầu vào:** 
    - img\_1d
    - centroids
    - labels: các labels chưa được gán.
    - k\_cluster: số màu muốn nén.
  + Đầu ra:
    - labels: Các labels sau khi được gán.
  + Đầu tiên, gán tất cả các labels bằng 0.
  + Sau đó sử dụng hàm **compute\_distance** để tính toán khoảng cách, thực hiện đánh dấu các labels sau đó trả về giá trị labels.
* **Hàm update\_centroids:** 
  + Đầu vào:
    - img\_1d
    - centroids: tập centroids chưa được cập nhật
    - labels: các labels đã được gán
    - k\_clusters:
  + Đầu ra:
    - Các centroids đã được cập nhật
  + Lấy các giá trị đã được gán nhãn và thực hiện tính lại tập centroids mới.
* **Hàm has\_converged:**
  + Đầu vào:
    - centroids
    - new\_centroids
  + Đầu ra:
    - True/False
  + Thực hiện kiểm tra xem tập mới có centroids mới có trùng với tập cũ hay không, nếu trùng sẽ trả về True và dừng vòng lặp kmeans.

1. **Kết quả:**
   1. **Ảnh gốc:**
   2. **Ảnh sau khi được nén với k = 3:**

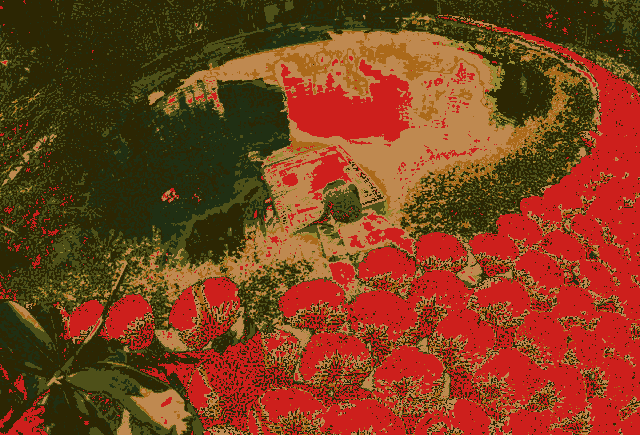
in pixels

random

* 1. **Ảnh sau khi được nén với k = 5:**

in pixels

random

* 1. **Ảnh sau khi được nén với k = 7:**

in pixels

random

1. **Nhận xét:**

* Trong cả 6 hình, đều một phần thể hiện được nội dung của anh gốc, tuy nhiên không còn rõ ràng, mức độ rõ ràng tăng dần theo (k = 3, 5, 7), tuy nhiên nội dung của những ảnh có k = 3 thì không thể dễ dàng nhận biết được khi so với ảnh gốc, chứng tỏ việc thực hiện thuật toán đã có một phần thành công.
* Trong những bức có đầu vào là “random”, nội dung thể hiện rõ rang hơn so với in\_pixels.

1. **Tài liệu tham khảo:**

**Geekforgeek:** [**https://www.geeksforgeeks.org/image-compression-using-k-means-clustering/**](https://www.geeksforgeeks.org/image-compression-using-k-means-clustering/)

**Simple Coding:**[**https://theironns.blogspot.com/2017/01/machine-learning-k-means-gioi-thieu-ve.html?m=1&fbclid=IwAR2qW1ochLSGp2S2ZDqVl-ZkeyUlzYAuYrFiT3THAcMaqioLTMlEpM0gPV4**](https://theironns.blogspot.com/2017/01/machine-learning-k-means-gioi-thieu-ve.html?m=1&fbclid=IwAR2qW1ochLSGp2S2ZDqVl-ZkeyUlzYAuYrFiT3THAcMaqioLTMlEpM0gPV4)

**Numpy:** [**https://numpy.org/doc/stable/reference/random/generated/numpy.random.choice.html**](https://numpy.org/doc/stable/reference/random/generated/numpy.random.choice.html)

**Funda:** [**https://machinelearningcoban.com/2017/01/01/kmeans/**](https://machinelearningcoban.com/2017/01/01/kmeans/)