**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**ĐẠI HỌC QUỐC GIA TP.HCM**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

­­­­­🙢🕮🙠 ­­­­­



**TOÁN ỨNG DỤNG VÀ THỐNG KÊ CHO CNTT**

**Project 1: Color Compression**

**Giảng viên hướng dẫn**: Vũ Quốc Hoàng, Nguyễn Văn Quang Huy,   
Ngô Đình Hy, Phan Thị Phương Uyên

**Họ tên**: Phạm Phú Toàn

**MSSV**: 21127183

**Lớp**: 21CLC08

Mục Lục

[**I.** **Mô tả các hàm** 3](#_Toc140430313)

[Nhập/xuất, tương tác: 3](#_Toc140430314)

[Thuật toán kmeans: 3](#_Toc140430315)

[**II.** **Nhận xét kết quả chạy:** 4](#_Toc140430316)

[Ảnh 1: dung lượng: 111 KB, 300x427, chế độ random 4](#_Toc140430317)

[Ảnh 2: dung lượng: 89.3 KB, 640x359, chế độ inpixel 6](#_Toc140430318)

[Ảnh 3: dung lượng: 140 KB, 1280x960, chế độ inpixel, kclusters = 5 8](#_Toc140430319)

[**III.** **Tham khảo** 8](#_Toc140430320)

1. **Mô tả các hàm**

## Nhập/xuất, tương tác:

* **Input(path):** 
  + Input: path của ảnh
  + Output: array đại diện cho ảnh
* **Output(picture):** 
  + Input: array ảnh
  + Ouput: hiện ảnh lên console
* **Interact():**
  + Output: các thông tin người dùng nhập vào: path – đường dẫn tói ảnh, clusters, type (random/inpixel), fileSave – tên file lưu
* **saveFile(fileSave, array):**
  + Input: là path/tên file muốn lưu, array ảnh đã qua xử lý còn k màu
  + Mô tả: lưu ảnh đã qua xử lý dưới định dạng và tên file mong muốn

## Thuật toán kmeans:

**Ý tưởng:**

+ Tạo ra k điểm centroids ban đầu.

+ Sau mỗi lần lặp ta sẽ nhóm các điểm ảnh (pixel) thành các vùng xung quanh centroids dựa trên khoảng cách của điểm ảnh đến centroids gần nó nhất - ta coi các kênh màu của một pixel là các vector trong không gian ba chiều (Euclide).

+ Thuật toán sẽ dừng sau khi lặp tối đa max\_iter lần hoặc khoảng cách của các centroids sau khi cập nhật là không thay đổi nhiều (<1).

* **kmeansInitCentroid(init\_centroids, k\_clusters, array):**
  + Input:
    - loại centroids muốn tạo – random/inpixel
    - số centroids/clusters sẽ tạo ra
    - nếu là inpixel sẽ lấy điểm ảnh ngẫu nhiên từ array ảnh
  + Output:
    - k điểm centroids tạo ra
* **labelElements(array, centroids):**
  + Vai trò:
    - Phân vùng các điểm ảnh theo k centroids
  + Output:
    - Trả về mảng labels kích thước bằng array, labels[i] đại diện cho centroids/clusters mà array[i] thuộc về
* **updateCentroids(labels, array, k\_cluster):**
  + Vai trò:
    - Update lại các centroids mới
  + Output:
    - một mảng k\_cluster centroids mới
* **kmeansCaculateDistance(X,Centroid):**
  + Vai trò:
    - Hàm tính khoảng cách từ một pixel đến centroids
  + Output:
    - Khoảng cách từ X đến Centroid – số thực
* **isTheSameCentroids(center1, center2):**
  + Vai trò:
    - Kiểm tra xem hai điểm (center1, center2) có giống nhau không bằng cách kiểm tra khoảng cách giữa chúng **(khoảng cách < 1)** thì ta coi là hai điểm giống nhau
* **returnLabels(labels, centroids):**
  + Vai trò:
    - Trả về một array ảnh với các điểm ảnh được thay thế bằng các centroids mà nó thuộc về
  + Output:
    - Array ảnh đã qua xử lý của thuật toán kmeans
* **kmeans(img\_1d, k\_clusters, max\_iter, init\_centroids='random'):**
  + Vai trò:
    - Hàm xử lý chính của thuật toán kmeans, gắn kết các hàm trên và chứa vòng lặp chính của thuật toán
  + Output:
    - Array ảnh đã qua xử lý – array từ hàm **returnLabels**, và k điểm centroids sau quá trình chạy thuật toán

1. **Nhận xét kết quả chạy:**

## Ảnh 1: dung lượng: 111 KB, 300x427, chế độ random

**kclusters = 3:**

A group of people posing for a picture

Description automatically generated

Thời gian chạy: 14s

Dung lượng ảnh nén: 15.2 KB

Đánh giá: vẫn nhận viết được chủ thể chính của bức ảnh

**kclusters = 5**

A person and two girls holding flowers

Description automatically generated

Thời gian chạy: 39s

Dung lượng ảnh nén: 25.6 KB

Đánh giá: Ảnh rõ nét hơn, các chi tiết phụ bắt đầu có thể nhận diện được

**kclusters = 7**

A person and two girls holding flowers

Description automatically generated

Thời gian chạy: 50s - 75s (2 lần chạy)

Dung lượng ảnh nén: 33.6 - 36.6 KB (2 lần chạy)

Đánh giá: Các chi tiết phụ như tay, quần áo, nơ… rõ nét hơn, tùy từng lần chạy mà màu sắc có sát với ảnh hơn.

## Ảnh 2: dung lượng: 89.3 KB, 640x359, chế độ inpixel

**kclusters = 3:**

A group of white clouds

Description automatically generated

Thời gian chạy: 19s

Dung lượng ảnh nén: 5.87 KB

Đánh giá: nhận biết được chủ thể chính của bức ảnh

**kclusters = 5**

A white clouds on a blue background

Description automatically generated

Thời gian chạy: 31s

Dung lượng ảnh nén: 8.62 KB

Đánh giá: Các mảng sáng tối xuất hiện rõ hơn

**kclusters = 7**

A white clouds on a blue background

Description automatically generated

Thời gian chạy: 89s (1 lần chạy)

Dung lượng ảnh nén: 10.2 KB

Đánh giá: các mảng sáng tối xuất hiện rõ hơn và tách bạch giữa các chủ thể chính rõ hơn

## Ảnh 3: dung lượng: 140 KB, 1280x960, chế độ inpixel, kclusters = 5



Thời gian chạy: 4p16 = 256s (1 lần chạy)

Dung lượng ảnh nén: 145 KB

Đánh giá: chi tiết ảnh tốt, chủ thể rõ ràng, các chi tiết gần thể hiện ổn

→ **Nhận xét chung:**

Thời gian thuật toán chạy thay đổi khá nhiều trên mỗi lần chạy, k càng tăng thì chất lượng ảnh càng rõ nhưng dung lượng cũng nặng hơn (dung lượng ảnh/chất lượng ảnh thay đổi mỗi lần chạy dù là k giống nhau – do random).

Chất lượng ảnh biểu diễn được chủ thể rõ ràng, mảng sáng tối và các chi tiết nhỏ biểu diển ở mức nhìn thấy được và có thể chấp nhận.

1. **Tham khảo**

Tân, T. (n.d.). *Youtube*. Retrieved from Thuật toán K-Means | KMean - KMeans clustering | Khai phá dữ liệu | K-Means Algorithm: https://www.youtube.com/watch?v=bIioxQE1ow4

Vũ Hữu Tiệp. (n.d.). *Machine Learning cơ bản*. Retrieved from Bài 4: K-means Clustering: https://machinelearningcoban.com/2017/01/01/kmeans/