Vietnam National University – Ho Chi Minh city

University of Science

Faculty of Information Technology

BÁO CÁO ĐỒ ÁN TOÁN ỨNG DỤNG VÀ THỐNG KÊ

APPLIED MATHEMATICS AND STATISTICS

Ho Chi Minh, 2021

Vietnam National University – Ho Chi Minh city

University of Science

Faculty of Information Technology

BÁO CÁO ĐỒ ÁN 2

2020 – 2021

APPLIED MATHEMATICS AND STATISTICS

Lớp: 19CLC7
Giáo viên hướng dẫn: Phan Thị Phương Uyên

STT	MSSV	Họ tên	Email
1	19127017	Trương Gia Đạt	19127017@student.hcmus.edu.vn

Ho Chi Minh, 2021

Table of Contents

Mẫu ảnh được sử dụng trong đồ án (512 x 512)

Một vài thông số cơ bản:

- Kích thước ảnh là 512 x 512 x 3.
- Channel(s): 3 tương ứng với R G B. $(0 \leq R, G, B \leq 255)$
- Tổng có 786.432 pixels.
- Kiểu dữ liệu số: $uint8 \rightarrow unsigned\ int\ 8 - bit = [0, 2^8 - 1] = [0, 255]$

Thay đổi độ sáng cho ảnh

- Ý tưởng:

$$g(i,j) = f(i,j) + \beta$$

  - $f(i,j)$ là giá trị các pixels của ảnh gốc.
  - β là một số nguyên dương để tăng độ sáng cho ảnh. $0 \leq \beta \leq 100$
  - $g(i,j)$ là giá trị các pixels mới sau khi tăng độ sáng cho ảnh.

- Kiểm tra điều kiện:
  - Tạo biến $threshold = 255 - \beta$.
  - Kiểm tra các giá trị trong RGB có lớn hơn threshold không?. Nếu có thì gán bằng 255 ngược lại thì cộng thêm giá trị β.
  - Ví dụ cho $\beta = 55$ thì $threshold = 255 - \beta = 255 - 55 = 200$.
  - Xét một giá trị 200 thì $200 \leq threshold$ nên lấy $200 + 55 = 255$.
  - Xét một giá trị 201 thì $201 > threshold$ nên khi lấy $201 + 55 = 256$ vượt mức lớn nhất cho phép là 255 nên gán giá trị đó bằng 255.

- Output:

Thay đổi độ tương phản cho ảnh

- Ý tưởng:

$$g(i,j) = \alpha . f(i,j)$$

  - $f(i,j)$ là giá trị các pixels của ảnh gốc.
  - α là một số thực để tăng độ tương phản cho ảnh. $1.0 \leq \alpha \leq 3.0$
  - $g(i,j)$ là giá trị các pixels mới sau khi tăng độ tương phản cho ảnh.

- Kiểm tra điều kiện:
  - Kiểm tra các giá trị trong RGB khi nhân với α có lớn hơn 255 không?. Nếu có thì gán bằng 255 ngược lại thì nhân với α.

- Output:

Thay đổi ảnh RGB thành ảnh xám

- Ý tưởng:
  - Đưa các giá trị điểm ảnh về gần giá trị 255 thì ảnh sẽ xám đi.
  - Mỗi channel là gồm nhiều ma trận các điểm ảnh. 3 channels thì gồm các ma trận các điểm ảnh R G B.
  - Lấy ma trận các điểm ảnh của R B G ra.
  - Khởi tạo biến $\boldsymbol{gray\_scaler = 0.2989 * R + 0.5870 * G + 0.1140 * B.}$
  - Thay đổi toàn bộ các ma trận của R B G thành $\boldsymbol{gray\_scaler.}$
- Output:

Lật ảnh (ngang – dọc)

1. **Lật ảnh ngang**

- Ý tưởng:
  - Thuộc tính **Numpy Shape** trả về các giá trị gồm row(height), column(width), color(…).
  - Lật ngang (trục Ox) nên ta sẽ xứ lý kích thước của row(height).
  - Duyệt $\boldsymbol{i \leftarrow 0}$ đến $\boldsymbol{shape[0]/2}$, chỉ cần xử lý một nửa kích thước của height.
  - Lấy từng pixel của hàng thứ $\boldsymbol{i}$ đổi với hàng thứ $\boldsymbol{shape[0] - i - 1}$ lẫn nhau và duyệt đến khi kết thúc.

- Output:

2. Lật ảnh dọc

- Ý tưởng:
  - Thuộc tính **Numpy Shape** trả về các giá trị gồm row(height), column(width), color(…).
  - Lật dọc (trục Oy) nên ta sẽ xử lý kích thước của column(width).
  - Duyệt $\boldsymbol{i \leftarrow 0}$ đến $\boldsymbol{shape[1]/2}$, chỉ cần xử lý một nửa kích thước của width.
  - Lấy từng pixel của cột thứ $\boldsymbol{i}$ đổi với cột thứ $\boldsymbol{shape[1] - i - 1}$ lẫn nhau và duyệt đến khi kết thúc.

- Output:

Chồng 2 ảnh cùng kích thước – Chỉ làm trên ảnh xám

- Ý tưởng:
  - Chỉnh lại kích thước (nếu cần) của hình 2 bằng với kích thước của hình gốc.
  - Đơn giản chỉ việc cộng hai ma trận lại với nhau.

- Kiểm tra điều kiện:
  - Do khi cộng 2 ma trận thì chắc chắn sẽ xảy tình trạng giá trị pixel > 255. Và khi lớn hơn 255 mà ta tự động đưa về 255 thì ảnh không còn màu xám mà chỉ toàn trắng hoặc ảnh bị hư.
  - Tạo 2 biến α, β $sao\ cho$ $\begin{cases} 0.0 \leq \alpha, \beta \leq 1.0 \\ \alpha + \beta = 1.0 \end{cases}$
  - Lấy ma trận của ảnh gốc nhân với α cộng với ma trận của ảnh 2 nhân với β.

- Hình 2:

- Output:

Làm mờ ảnh

- Ý tưởng:
  - Sử dụng kernel theo Gaussian

$$kernel = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

  - Áp dụng hướng tiếp cận Convolution để tạo ra ma trận có các giá trị điểm ảnh mới.

$$\begin{bmatrix}a_{11} & a_{12} & a_{13} & a_{14} & a_{15}\\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25}\\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35}\\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45}\\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55}\end{bmatrix} (1)$$

  - Tạo một ma trận có kích thước cùng với ***kernel*** đóng vai trò là một ***mirror*** và lấy ma trận vừa tạo nhân với ma trận ***kernel***.

$$\begin{bmatrix}a_{11} & a_{12} & a_{13}\\ a_{21} & a_{22} & a_{23}\\ a_{31} & a_{32} & a_{33}\end{bmatrix} * \begin{bmatrix}0.0625 & 0.125 & 0.0625\\ 0.125 & 0.25 & 0.125\\ 0.0625 & 0.125 & 0.0625\end{bmatrix}$$

    thì tại ví trí $\boldsymbol{a_{22}}$ của ma trận (1) sẽ có giá trị mới là $a_{22} = a_{11} * 0.0625 + a_{12} * 0.125 + a_{13} * 0.0625 + a_{21} * 0.125 + a_{22} * 0.25 + a_{23} * 0.125 + a_{31} * 0.0625 + a_{32} * 0.125 + a_{33} * 0.0625$.

  - Dịch ***mirror*** sang một cột tiếp theo và khi đến cột cuối của ma trận gốc thì đưa ma trận ***mirror*** xuống dòng mới của cột đầu tiên và tính toán tương tự cho đến hết ma trận gốc.

  - Một ma trận mới được tạo ra chứa các giá trị của một bức ảnh mờ.

- Output: