Bài thực hành 1 - Xử lý ảnh

Lớp N01 - Năm học 2021/2022

August 2021

1 Hướng dẫn

- Tải file ảnh cần thiết để thực hành tại:

  `https://people.math.sc.edu/Burkardt/data/tif/cameraman.tif`

- Mở MATLAB và tạo một thư mục mới để làm việc. Đưa file ảnh vừa tải vào thư mục mới đó.
- Sau khi hoàn thành bài tập, lưu kết quả mỗi bài tập của phần 2.2, 3.2 và 4.2 (mỗi câu hỏi sẽ có hướng dẫn cụ thể) vào một báo cáo (định dạng docx hoặc PDF) và nộp bài tập của mình trên Teams. Sinh viên nhớ kiểm tra đầy đủ các thao tác nộp bài.
- Sinh viên làm bài thực hành theo từng cá nhân.
- Hạn nộp bài: 23h59' ngày 05/09/2021. Các trường hợp nộp muộn hoặc sao chép bài làm của sinh viên khác coi như không nộp bài.

2 Biểu diễn ảnh trong MATLAB

2.1 Nhắc lại kiến thức

Một bức ảnh độ xám g có kích thước $M \times N$ được biểu diễn bởi một ma trận $M \times N$ chứa giá trị của các điểm ảnh. Mỗi giá trị $g(m, n)$ là độ xám của điểm ảnh tại vị trí (m, n) trong đó: m là số hàng và n là số cột. Điểm đầu tiên được đánh dấu là điểm tại góc trên bên trái của ảnh. Với MATLAB, chỉ số của các phần tử trong ma trận được đánh số từ $g(1, 1)$.

$$g = \begin{bmatrix} g(1,1) & \cdots & g(1,N) \\ \vdots & \ddots & \vdots \\ g(M,1) & \cdots & g(M,N) \end{bmatrix}$$

Để lưu ảnh trong bộ nhớ, mỗi điểm ảnh sẽ được lưu trữ bởi một số lượng bits b. Khi $b = 1$ mỗi điểm ảnh có thể nhận hai giá trị 0 hoặc 1. Tương tự như

vậy với $b = 8$ bits (1 byte), chúng ta có 256 mức độ xám khác nhau cho mỗi điểm ảnh (từ 0 đến 255). Giá trị 0 thường được sử dụng cho màu đen và giá trị lớn nhất (1 đối với ảnh 1 bit/pixel và 255 với ảnh 8 bits/pixel) tương đương với màu trắng.

Ví dụ ảnh đen trắng:

$$g_1 = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

Ảnh độ xám 8-bits:

$$g_2 = \begin{bmatrix} 19 & 20 & 66 \\ 251 & 71 & 170 \\ 120 & 50 & 99 \end{bmatrix}$$

Để hiển thị ảnh trên MATLAB, chúng ta có thể chuẩn hoá ảnh sao cho giá trị mỗi điểm ảnh nằm trong khoảng 0-1. Để làm được điều này chỉ cần chia giá trị mỗi điểm ảnh cho $2^b - 1$ (là giá trị lớn nhất mà một điểm ảnh có thể nhận). Hình ảnh sau đó được gọi là hình ảnh **đã được chuẩn hoá**:

Ví dụ khi chuẩn hoá g_1 ta được:

$$g_1^{'} = \frac{g_1}{1} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

Khi chuẩn hoá g_2 ta được:

$$g_2^{'} = \frac{g_2}{255} = \begin{bmatrix} 0.0745 & 0.0784 & 0.2588 \\ 0.9843 & 0.2784 & 0.6667 \\ 0.4706 & 0.1961 & 0.3882 \end{bmatrix}$$

2.2 Bài tập

1. Tạo một ảnh 8-bits $X1$ có kích cỡ 200×300 chỉ gồm các điểm ảnh màu đen. Đưa code MATLAB và ảnh vào báo cáo.

   Gợi ý: Để tạo ma trận trong MATLAB chúng ta có thể dùng một trong các câu lệnh sau:

   ```
   X = zeros(M,N); % Tao mot ma tran co M x N chua
       cac phan tu co gia tri 0, trong do M la so
       hang va N la so cot.
   X = ones(M,N); % Tao mot ma tran co M x N chua
       cac phan tu co gia tri 1.
   X = ones(M,N)*50; % Tao mot ma tran co M x N chua
        cac phan tu co gia tri 50.
   ```

   Để hiển thị và lưu ảnh trong MATLAB chúng ta có thể dùng các câu lệnh sau (chú ý khi hiển thị chúng ta nên sử dụng ảnh đã chuẩn hoá):

```
figure % Tao mot do thi moi
imshow(X/255) % X la mot anh 8-bits chua chuan
    hoa
imsave
```

2. Tạo một ảnh 8-bits có kích cỡ 200x300 chưa toàn các điểm ảnh màu trắng. Đưa code MATLAB và ảnh vào báo cáo.

3. Tạo một ảnh 8-bits có kích cỡ 200x300 chưa toàn các điểm ảnh màu xám sáng (sinh viên tự chọn giá trị màu). Đưa code MATLAB và ảnh vào báo cáo.

4. Tạo một ảnh 8-bits có kích cỡ 200x300 chưa toàn các điểm ảnh màu xám đậm (sinh viên tự chọn giá trị màu). Đưa code MATLAB và ảnh vào báo cáo.

5. Tạo một ảnh 8-bits có kích cỡ 200x300 có nền màu xám sáng và có một đường màu đen nằm ngang có độ dày 10 pixels (Vị trí của đường màu đen do sinh viên tự chọn). Đưa code MATLAB và ảnh vào báo cáo.

   Gợi ý: Để sửa giá trị trong một ma trận với MATLAB có thể dùng các câu lệnh:

   ```
   X(3,10) = 0; % Sua gia tri tai hang 3 cot 10
   X(4,:) = 0; % Sua tat ca gia tri tai hang 4
   X(:,5) = 0; % Sua tat ca gia tri tai cot 5
   X(4:9,:) = 0; % Sua tat ca gia tri tai cac hang
       tu 4 den 9
   X(:,3:8) = 0; % Sua tat ca gia tri tai cot tu 3
       den 8
   X(5:9,6:10) = 0; % Sua tat ca gia tri nam dong
       thoi tai hang 5 den 9 va cot 6 den 10
   ```

6. Tạo một ảnh 8-bits có kích cỡ 200x300 có nền màu xám đậm, có một đường màu đen nằm ngang có độ dày 10 pixels và một đường màu trắng nằm dọc có độ dày 10 pixels. Các điểm nằm trong vùng giao nhau giữa đường đen và trắng thì giữ nguyên màu xám đậm như màu nền. Đưa code MATLAB và ảnh vào báo cáo.

7. Tạo một ảnh 8-bits có kích cỡ 200x300 có nền màu xám sáng và có một ô màu đen nằm trong hình có kích cỡ 30x30. Đặt ô màu đen đó vào chính giữa hình. Đưa code MATLAB và ảnh vào báo cáo.

3 Nhiễu trong ảnh

3.1 Nhắc lại kiến thức

Khi chụp ảnh, truyền tải ảnh hoặc nén ảnh, ảnh có thể bị giảm chất lượng. Trong phần này chúng ta tập trung đến hai loại giảm chất lượng thường gặp

trong ảnh:

- **Nhiễu Gausse**: Là loại nhiễu ảnh hưởng đến tất cả các điểm ảnh. Mỗi điểm ảnh sẽ là tổng của giá trị thực của điểm ảnh và một giá trị nhiễu. Giá trị nhiễu này được phân bố theo hàm Gausse. Hàm phân bố này được mô tả bởi phương sai v và trung bình m.
- **Nhiễu muối tiêu**: Là loại nhiễu chỉ ảnh hưởng đến một số lượng nhất định điểm ảnh. Với ảnh độ xám, nhiễu này tương ứng với việc đặt các điểm ảnh có độ xám cực tiểu (màu đen) hoặc độ xám cực đại (màu trắng) vào ảnh một cách ngẫu nhiên với một tần suất p. Giá trị p càng lớn thì ảnh bị nhiễu càng nhiều.

Để thêm nhiễu vào ảnh trong MATLAB chúng ta dùng các câu lệnh sau (chú ý ảnh đầu vào là ảnh đã được chuẩn hoá):

```
Y = imnoise(X, 'gaussian', m, v); % Them nhieu Gausse co
    trung binh m va phuong sai v vao anh
Y = imnoise(X, 'salt & pepper', p); % Them nhieu muoi
    tieu voi ti le p vao anh
```

3.2 Bài tập

1. Mở file ảnh `cameraman.tiff`, lưu vào ma trận $X1$. Chuẩn hoá ma trận $X1$. Đưa code MATLAB vào báo cáo.

   Để đọc ảnh trong MATLAB chúng ta dùng các câu lệnh sau:

   ```
   X = imread('image.bmp') % Doc file image.bmp
   X = double(X) % Chuyen du lieu ve dang double de
       chuan hoa
   ```

2. Thêm vào $X1$ một nhiễu Gausse với trung bình 0 và phương sai 0.01 và lưu ảnh mới vào ma trận $X2$. Đưa code và ảnh $X2$ vào báo cáo. Thử thay đổi phương sai xem có ảnh hưởng gì đến chất lượng ảnh?

3. Thêm vào $X1$ một nhiễu muối tiêu với tỉ lệ $p = 0.05$ và lưu vào ma trận $X3$. Đưa code và ảnh $X3$ vào báo cáo. Thử thay đổi p xem có ảnh hưởng gì đến chất lượng ảnh.

4. Vẽ trên một đồ thị giá trị các điểm ảnh tại hàng 128 của X1, X2, X3. Đưa code và đồ thị vào báo cáo. Nhận xét về đồ thị thu được.

   Gợi ý: Để vẽ một biểu đồ gồm nhiều đường (trong đó mỗi đường là một chuỗi các giá trị), chúng ta có thể sử dụng câu lệnh:

   ```
   figure
   plot(1:N, x1)
   hold on
   plot(1:N, x2, 'g')
   plot(1:N, x3, 'r')
   ```

4 Bộ lọc

4.1 Nhắc lại kiến thức

Bộ lọc là một hệ 2D để chuyển đổi một ảnh đầu vào thành một ảnh đầu ra. Có hai kiểu bộ lọc:

- **Lọc tuyến tính**: (tương ứng với hệ tuyến tính bất biến không gian LSI) là một phép tích chập 2D giữa ảnh đầu vào và một bộ lọc $h(m, n)$ có kích cỡ $M_h \times N_h$ gọi là **mặt nạ** của phép tích chập (thông thường $M_h = N_h$). Khi sử dụng bộ lọc tuyến tính mỗi điểm ảnh của ảnh đầu ra được tính bằng trung bình có trọng số của các điểm ảnh lân cận trong ảnh đầu vào. Mặt nạ h chính là trọng số của phép tính toán đó.
- **Lọc phi tuyến**: được sử dụng để lọc một số loại nhiễu đặc biệt. Phép tính trong loại bộ lọc này thường là các phép toán phi tuyến tính như phép lấy trung vị.

Để tạo bộ lọc h trong MATLAB chúng ta có thể tạo ma trận ở dạng bình thường, hoặc sử dụng hàm `fspecial`.

```
h = ones(3,3)/9; % Bo loc trung binh 3 x 3
h = [1 0 1; 0 2 0; 1 0 1]
h = fspecial('average', [3 3]); % Bo loc trung binh 3 x 3
h = fspecial('gaussian', [5 5], 1); % Bo loc gausse 5x5
    voi do lech chuan 1
```

Để đưa ảnh đầu vào X qua bộ lọc h chúng ta sử dụng hàm `imfilter` (chú ý ảnh đầu vào X là ảnh đã được chuẩn hoá):

```
Y = imfilter(X, h, 'replicate');
```

Để sử dụng bộ lọc trung vị, chúng ta gọi hàm `medfilt2` (chú ý ảnh đầu vào X là ảnh đã được chuẩn hoá):

```
Y = medfilts(X, [3 3]); % Loc trung vi 3 x 3
```

4.2 Bài tập

1. Áp dụng bộ lọc trung bình 3×3 với ảnh $X2$ ở phần trước và lưu vào ma trận $Y2$. Đưa code và ảnh $Y2$ vào báo cáo. Nhiễu của ảnh $X2$ có được lọc hiệu quả không?

2. Áp dụng bộ lọc trung vị 3×3 với ảnh $X3$ ở phần trước và lưu vào ma trận $Y3$. Đưa code và ảnh $Y3$ vào báo cáo. Nhiễu của ảnh $X3$ có được lọc hiệu quả không?

3. Để kiểm tra chất lượng ảnh sau khi khử nhiễu, người ta thường sử dụng một thông số là PSNR (Peak Signal to Noise Ratio) - tỉ số tín hiệu cực đại trên nhiễu. PSNR được tính theo công thức:

$$\text{PSNR} = 10 \log_{10} \frac{R^2}{\frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} (x(m,n) - y(m,n))^2}$$

Trong đó x và y là các điểm ảnh của ảnh ban đầu và ảnh sau khi khử nhiễu, R là giá trị tối đa có thể của một pixel. Giá trị PSNR càng cao thì ảnh được coi là càng có chất lượng tốt so với ảnh gốc.

PSNR có thể được tính toán đơn giản bằng hàm có sẵn của MATLAB với X là ảnh ban đầu và Y là ảnh đã khử nhiễu:

```
PSNR = psnr(Y, X);
```

Tính PSNR với hai phép khử nhiễu ở câu 2 và 3 (tương đương với việc tính PSNR của Y2, Y3 với X1). Đưa giá trị của PSNR vào báo cáo.

4. Thử khử nhiễu ảnh $X2$ và $X3$ với các bộ lọc sau:

   - Lọc trung bình với kích thước bộ lọc lần lượt là: $3 \times 3, 5 \times 5, 7 \times 7$.
   - Lọc Gausse với độ lệch chuẩn lần lượt là 2, 1.5, 1, 0.5.
   - Lọc trung vị với kích thước bộ lọc lần lượt là: $3 \times 3, 5 \times 5, 7 \times 7$.

   Đưa giá trị PSNR của mỗi lần thử vào báo cáo dưới dạng bảng.