Xử lý ảnh

Chương 2: Làm việc với không gian màu

Giảng viên: Đỗ Thanh Hà *VNU* - University of Science

1.1 Bài tập 1: Biến đổi không gian màu

Cho ảnh màu $I \in \mathbb{R}^{m \times n \times 3}$ (lenna_color.jpg). Yêu cầu:

1. Đọc và hiển thị ảnh I trong không gian màu RGB.
2. Trích xuất và hiển thị riêng biệt ba kênh màu R, G, B.
3. Chuyển đổi ảnh từ không gian màu RGB sang không gian màu HSV.
4. Hiển thị riêng biệt ba kênh H, S, V của ảnh HSV.

Tiếp theo, với ảnh đã được chuyển đổi sang không gian màu HSV:

1. Đọc ảnh I' trong không gian màu HSV và hiển thị các kênh H, S, V.
2. Thực hiện chuyển đổi ảnh từ không gian màu HSV ngược lại không gian màu RGB.
3. So sánh ảnh thu được với ảnh gốc I, đánh giá sự thay đổi về màu sắc và độ chính xác của phép biến đổi.

1.2 Bài tập 2: Biến đổi tuyến tính màu

Trong xử lý ảnh, phép **biến đổi tuyến tính màu** được sử dụng để hiệu chỉnh màu sắc của ảnh, giúp đảm bảo màu sắc hiển thị chính xác hơn hoặc thay đổi phong cách màu của ảnh. Phép biến đổi này được biểu diễn dưới dạng ma trận A có kích thước 3×3, ánh xạ màu từ ảnh đầu vào J (ảnh sai màu) sang ảnh gốc I (ảnh chuẩn). Phép biến đổi này được mô tả bởi phương trình:

$$Y = A^{-1}X \tag{1.1}$$

Với:

- X là ma trận $3 \times n$ chứa n điểm màu từ ảnh sai màu J.
- Y là ma trận $3 \times n$ chứa n điểm màu từ ảnh gốc I.
- A^{-1} là ma trận hiệu chỉnh màu cần tìm.

Yêu cầu thực hiện

1. **Đọc và hiển thị hai ảnh** I (images.jpg) và J (images2.jpg).

2. **Chọn ít nhất 3 điểm màu** từ ảnh I và ảnh J (các điểm tương ứng).
   - Các điểm này nên phân bố đều trong không gian màu, bao gồm vùng sáng, vùng tối và vùng trung tính.
   - Ghi lại giá trị RGB của các điểm màu tương ứng.
3. **Tính toán ma trận hiệu chỉnh màu** A bằng cách giải hệ phương trình tuyến tính:
$$B = A^{-1} = YX^{-1} \tag{1.2}$$
4. **Áp dụng ma trận A lên toàn bộ ảnh J để tạo ra ảnh đã hiệu chỉnh.**
5. **Hiển thị và so sánh ảnh sau khi hiệu chỉnh với ảnh gốc.**
6. **(Mở rộng)** Tăng số điểm mapping lên $n > 3$ và sử dụng phương pháp *Least Squares* để tìm ma trận A tối ưu hơn:
$$B = A^{-1} = YX^T(XX^T)^{-1} \tag{1.3}$$

1.3 Bài tập 3: Kỹ thuật ChromaKeying

Cho ảnh $I \in \mathbb{R}^{m\times n\times 3}$ (lion.jpg) và ảnh $J \in \mathbb{R}^{m\times n\times 3}$ (lionHSV.jpg). Thực hiện yêu cầu sau:

1. Đọc và hiển thị ảnh I, J lên cùng một khung hình.
2. Gọi 3 kênh màu tương ứng của J là J_h, J_s và J_v. Chuyển kiểu của J_h về kiểu `double`. Nhị phân hoá J_h bằng cách đặt những giá trị lớn hơn 0.22 và nhỏ hơn 0.45 bằng 1, các giá trị khác bằng 0. Gọi ảnh nhị phân thu được là B, hiển thị ảnh B lên màn hình.
3. Sử dụng đoạn mã giả sau nhằm xử lý phông nền màu xanh trên ảnh $I \in \mathbb{R}^{m\times n\times 3}$. Trong đó cột số 1, 2, 3, 4 trong tệp `data.txt` chứa thông tin về các vector v_1^T, v_2^T, v_3^T, và v_B.

   Hiển thị ảnh I, K trên cùng một khung hình.

1.4 Bài tập 4: Kỹ thuật Shadow Remove

Thực hiện các yêu cầu sau:

1. Cho ảnh $I(images.jpg)$ và $J(binary.png) \in R^{h\times w}$. Biến đổi ảnh I sao cho I và J có cùng kích thước. Chuyển kiểu của I về *double*. Gọi **dbI = I / 255**. Hiển thị dbI, J lên màn hình dưới mỗi ảnh là thông tin về kích thước của ảnh đó.

ảnh I	**ảnh J**
kích thước của ảnh **I**	kích thước của ảnh **J**

Algorithm 1 CHROMAKEYING

procedure ChromaKeying
  Tính id là vector cột chứa chỉ số của các phần tử khác 0 trong v_B
  for $b = 1$ to 3 **do**
    Gán các phần tử trong v_T^b có chỉ số id giá trị bằng 0
    $K_b = zeros(m, n)$
    for $j = 1$ to n **do**
      count = 1
      for $i = (j-1) \times m + 1$ to $j \times m$ **do**
        $K_b(count, j) = v_T^b[i]$
        count = count + 1
      end for
    end for
  end for
  Kết hợp 3 kênh màu K_b (b = 1, 2, 3) để tạo thành ảnh màu K
  Chuyển kiểu của K về kiểu $uint8$
end procedure

2. (a) Nhị phân hoá ma trận J sao cho pixel trắng có giá trị bằng 0 và pixel màu đen có giá trị bằng 1.

   (b) Đọc dữ liệu từ file **alpha.txt** và hiển thị hình ảnh.

3. Tìm ảnh SF sử dụng thuật toán sau. Hiển thị SF và ảnh I lên màn hình.

Algorithm 2 Thuật toán Shadow Remove

procedure ShadowRemove
  $\mathbf{r} = \begin{pmatrix} 1.2 \\ 1.0 \\ 0.7 \end{pmatrix}$
  $[m, n] = \text{size}(\alpha)$
  $\mathbf{I} \in \mathbb{R}^{m \times n}$ là ma trận toàn giá trị 1
  for $i = 1$ to 3 **do**
    $c^{(i)}$ là kênh màu thứ i của ảnh `dbl`
    $rc^{(i)} = \mathbf{I} \times \frac{r^{(i)}+1}{\alpha \times r^{(i)}+1} .* c^{(i)}$
  end for
  Kết hợp 3 kênh màu $rc^{(i)}$ để thu được ảnh SF
  Đặt giá trị SF > 1 bằng 1
end procedure

Trong đó, phép toán $.*$ và $./$ giữa hai ma trận $A = (a_{ij})_{m \times n}$, $B = (b_{ij})_{m \times n}$ được định nghĩa như sau:

$$C = A./B = (a_{ij}/b_{ij})_{m \times n}, \quad D = A. * B = (a_{ij} \times b_{ij})_{m \times n}$$