|  |  |
| --- | --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO** | **BỘ NÔNG THÔN VÀ PTNT** |
| **TRƯỜNG ĐẠI HỌC THỦY LỢI** | |



NGUYỄN DOANH CHÍNH

**HIỆU CHỈNH ÁNH SÁNG TRONG ẢNH**

**ĐỒ ÁN TỐT NGHIỆP**

**HÀ NỘI, NĂM 2023**

|  |  |
| --- | --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO** | **BỘ NÔNG THÔN VÀ PTNT** |
| **TRƯỜNG ĐẠI HỌC THỦY LỢI** | |

NGUYỄN DOANH CHÍNH

**HIỆU CHỈNH ÁNH SÁNG TRONG ẢNH**

|  |  |
| --- | --- |
| Ngành: | Hệ thống thông tin |
| Mã số: |  |

|  |  |
| --- | --- |
| Người hướng dẫn: | THS. Nguyễn Ngọc Quỳnh Châu |

**HÀ NỘI, NĂM 2023**

**CHƯƠNG 1: TỔNG QUAN LÝ THUYẾT**

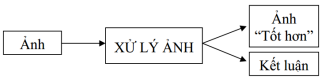
1. **Cơ sở lý thuyết**
   1. **Tổng quan xử lý ảnh**

Xử lý ảnh là một lĩnh vực nghiên cứu trong lĩnh vực thị giác và ứng dụng chủ yếu của khoa học máy tính, tập trung vào việc phân tích, biến đổi, và hiểu thông tin từ dữ liệu hình ảnh và quá trình biến đổi từ ảnh ban đầu sang một ảnh mới với các đặc tính tuân theo ý muốn người sử dụng. Mục tiêu chính của xử lý ảnh là chuyển đổi hình ảnh từ dạng thô sang dạng có ý nghĩa và hữu ích cho mục đích cụ thể. Các kỹ thuật xử lý ảnh bao gồm các phép toán cơ bản như biến đổi hình dạng, điều chỉnh độ sáng/tương phản, lọc ảnh, và các phép toán phức tạp hơn như nhận dạng đối tượng và trích xuất đặc trưng.

Quá trình xử lý ảnh được xem như là quá trình thao tác ảnh đầu vào nhằm cho ra kết quả mong muốn. Kết quả đầu ra của một quá trình xử lý ảnh có thể là một ảnh “tốt hơn” hoặc một kết luận.

Khi nói đến “ảnh tốt hơn”, chúng ta đang nói về việc cải thiện chất lượng hình ảnh thông qua các kỹ thuật như điều chỉnh độ sáng/tương phản, lọc ảnh, hoặc biến đổi hình dạng. Một ảnh có thể được coi là “tốt hơn” nếu nó rõ ràng hơn, có độ phân giải cao hơn, hoặc có màu sắc phong phú hơn so với ảnh gốc.

Một “kết luận” trong xử lý ảnh thường liên quan đến việc trích xuất thông tin quan trọng từ hình ảnh. Điều này có thể bao gồm việc nhận dạng đối tượng, phân loại hình ảnh, hoặc phát hiện mẫu.



Hình 1 :Quá trình xử lý ảnh

* 1. **Khái niệm ảnh**
     1. **Ảnh và Điểm ảnh (Picture Element)**

Ảnh tự nhiên, thường được biểu diễn như một hàm liên tục của không gian và độ sáng, đòi hỏi quá trình số hóa để trở thành một tập hợp điểm ảnh, hay còn gọi là pixel, đại diện cho vị trí và độ sáng tương ứng. Mỗi pixel là một điểm ảnh, và khoảng cách giữa chúng được thiết lập sao cho mắt người không phân biệt được sự chuyển động giữa các pixel khi nhìn bình thường.

Quá trình số hóa này thường được thực hiện thông qua mô hình toán học, trong đó ảnh có thể được xem như là một hàm hai biến f(x, y), với giá trị xám giới hạn từ 0 đến 255 trong trường hợp ảnh xám.

Điểm ảnh được xem như là dấu hiệu hay cường độ sáng tại một toạ độ trong không gian của đối tượng .

Ảnh được xem như là một tập hợp các điểm ảnh. Khi số hoá, nó thường được biểu diễn bởi bảng hai chiều I(n,p): n dòng và p cột. Ta nói ảnh gồm n x p điểm ảnh. Người ta thường kí hiệu I(x,y) để chỉ một điểm ảnh.

Mỗi pixel trong ảnh được coi như một dấu hiệu hoặc cường độ sáng tại một vị trí cụ thể trong không gian của đối tượng. Việc giới hạn giá trị của hàm f(x, y) trong khoảng 0 đến 255 giúp đảm bảo rằng mỗi điểm ảnh có thể được biểu diễn bằng một byte. Đối với ảnh màu, mỗi màu (R, G, B) có thể được mô tả thông qua các hàm riêng biệt của (x, y).

Số điểm ảnh tạo nên một ảnh gọi là độ phân giải (resolusion). Độ phân giải thường được biểu thị bằng số điểm ảnh theo chiều dọc và chiều ngang của ảnh. Ảnh có độ phân giải càng cao càng rõ nét. Như vậy, ảnh càng to thì càng bị vỡ hạt, độ mịn càng kém. Ảnh có thể được biểu diễn theo mô hình Vector hoặc mô hình Raster.

* **Mô hình Vector và Mô hình Raster trong Biểu Diễn Ảnh:**
  + **Mô hình Raster:**

Đây là cách biểu diễn ảnh thông dụng nhất hiện nay,Mô hình Raster là một phương pháp biểu diễn ảnh bằng cách sử dụng một lưới các pixel. Mỗi pixel có độ sáng và màu sắc được xác định tại một vị trí cụ thể trên lưới. Cả ảnh được chia thành một số lượng hữu hạn các pixel, và mỗi pixel đại diện cho một điểm ảnh riêng lẻ. Tuỳ theo yêu cầu thực thế mà mỗi điểm ảnh được biểu diễn qua 1 hay nhiều bit. Mật độ điểm ảnh trên một đơn vị kích thước vật lý được gọi là độ phân giải. Ảnh có độ phân giải cao thì càng đẹp, càng mịn và càng thể hiện rõ chi tiết. Việc lựa chọn độ phân giải thích hợp tuỳ thuộc vào nhu cầu sử dụng và đặc trưng của mỗi ảnh cụ thể Đây là mô hình phổ biến trong xử lý ảnh và đồ họa máy tính,thuận lợ cho việc hiển thị và in ấn.

Ảnh raster thường được thu từ camera,các máy chiếu chụp quét… và chinh là đối tượng chinh của xử lý ảnh và thị giác máy tính

* + **Mô hình Vector:**

Ảnh vector là một loại ảnh được tạo thành từ các thành phần hình học đơn giản như điểm, đường thẳng, hình khối, v.v. Thay vì được lưu trữ dưới dạng ma trận điểm ảnh như ảnh raster, ảnh vector được biểu diễn dưới dạng tọa độ các thành phần trong ảnh.

Điều đặc biệt của ảnh vector là khả năng thay đổi kích thước mà không bị mất chất lượng, không xuất hiện hiện tượng răng cưa như ảnh raster. Điều này là do dữ liệu trong ảnh vector chỉ gồm các thông tin về tọa độ, không phụ thuộc vào độ phân giải như trong ảnh raster. Do đó, ảnh vector thường tiết kiệm dung lượng lưu trữ hơn

Tuy nhiên, màu sắc trong ảnh vector thường ít tinh tế hơn so với ảnh raster, vì chỉ được biểu diễn dưới dạng một số màu cơ bản hoặc các phần tử màu sắc đơn giản khác.

Thông thường, ảnh vector được sử dụng trong thiết kế logo, banner, giao diện đồ họa và các ứng dụng tương tự. Tuy nhiên, trong lĩnh vực xử lý ảnh và thị giác máy tính, thường không đề cập đến xử lý ảnh vector, mà tập trung vào xử lý ảnh raster.

* + 1. **Mức xám**
    - **Mức xám:**

Mức xám trong ảnh là kết quả của quá trình mã hóa cường độ sáng tương ứng của mỗi điểm ảnh bằng một giá trị số. Cách mã hóa thường sử dụng một số mức khác nhau, ví dụ như 16, 32, hoặc 64 mức. Tuy nhiên, mã hóa với 256 mức là phổ biến nhất do lợi ích kỹ thuật. Với 256 mức, mỗi điểm ảnh được biểu diễn bằng 8 bit, từ 0 đến 255.

Ảnh có chỉ hai mức xám được gọi là ảnh nhị phân, mỗi điểm ảnh chỉ có thể có giá trị 0 hoặc 1. Nếu ảnh có mức xám lớn hơn 2, nó được gọi là ảnh đa cấp xám hoặc ảnh màu, trong đó mỗi điểm ảnh có thể có giá trị mức xám khác nhau.

Với ảnh màu, mọi màu có thể được tạo ra từ ba màu cơ bản: Đỏ, Lục (Xanh lá cây), và Lam (Xanh dương). Mỗi điểm ảnh của ảnh màu được lưu trữ trong 3 byte, tương ứng với 24 bit, và do đó, có 224 = 16,7 triệu màu khả dụng. Ảnh xám, tuy nói là ảnh màu, thực chất chỉ sử dụng màu xám, có các thành phần R, G, B trong hệ thống màu RGB có cùng cường độ. Điều này có nghĩa là mỗi điểm ảnh sẽ có một mức xám xác định.

Mức xám không chỉ là cách biểu diễn sự đơn giản của ảnh, mà còn liên quan đến việc truyền đạt thông tin về cường độ sáng tại mỗi điểm ảnh. Trong quá trình số hóa, mức xám được ánh xạ từ giá trị cường độ sáng của điểm ảnh, và nó thường được biểu diễn dưới dạng giá trị số từ 0 đến 255 trong trường hợp ảnh xám.

Ảnh xám thường được coi như là biểu hiện của không gian hai chiều, trong đó mỗi điểm ảnh tương ứng với một giá trị cụ thể của hàm hai biến. Quá trình này là quan trọng để mắt người không phân biệt được sự chuyển động giữa các điểm ảnh khi nhìn bình thường.

Nếu nhìn sâu hơn vào ảnh xám, nó thực chất là một dạng đặc biệt của ảnh màu, trong đó cả ba thành phần màu (R, G, B) có giá trị bằng nhau. Điều này tạo ra hiệu ứng màu xám tại mỗi điểm ảnh.

Mức xám không chỉ quan trọng trong việc truyền đạt thông tin về sự sáng tối của ảnh, mà còn là cơ sở cho nhiều phương pháp xử lý ảnh, bao gồm điều chỉnh độ tương phản, làm sắc nét, phát hiện cạnh, và nhiều ứng dụng khác.

* + 1. **Ánh Số**

Ảnh số là một tập hợp hữu hạn các điểm ảnh, hay còn gọi là pixel. Mỗi pixel trong ảnh số có thể được biểu diễn dưới dạng một phần tử trong ma trận hai chiều.

Ảnh số cũng có thể được hiểu như một hàm hai chiều, f(x, y), trong đó x và y là tọa độ không gian và biên độ của f tại mỗi cặp tọa độ (x, y) biểu diễn cường độ hoặc mức độ màu xám của hình ảnh tại điểm đó.

Khi tất cả các giá trị x, y và cường độ của f đều là các đại lượng rời rạc, hữu hạn, chúng ta gọi đó là hình ảnh kỹ thuật số. Độ lớn (amplitude) của hàm f được gọi là độ sáng (intensity) hay độ xám (gray level) của ảnh tại điểm đó

Tùy vào giá trị dung để biểu diễn điểm ảnh, có thể chia ảnh số thành 2 loại : ảnh đen trắng và ảnh màu.

* **Ảnh đen trắng**

Ảnh đen trắng là loại ảnh chỉ bao gồm ảnh đen và ảnh trắng, không chứa bất kỳ màu nào khác. Thực tế ảnh đen trắng được phân loại thành 2 loại khác nhau là ảnh nhị phân và ảnh đa mức xám. Người ta phân mức đen trắng của ảnh thành L mức. Nếu L bằng 2 nghĩa là chỉ có 2 mức 0 và 1, hay còn được gọi là ảnh nhị phân. Nếu L lớn hơn 2 thì ta có ảnh đa cấp xám.

Nói cách khác, mỗi điểm ảnh nhị phân được mã hóa trên 1 bit còn ảnh xám 256 mức đã được mã hóa 8 bit (1byte).

* **Ảnh màu**

Ảnh màu là một dạng biểu diễn của ảnh số, trong đó mỗi pixel không chỉ biểu diễn một giá trị độ sáng, mà còn biểu diễn màu sắc. Mỗi pixel trong ảnh màu được biểu diễn bằng một bộ ba giá trị (r, g, b), tương ứng với mức độ của màu đỏ (red), xanh lục (green), và xanh lam (blue). [Khi kết hợp các màu này theo tỷ lệ nhất định, ta có thể tạo ra nhiều màu sắc khác nhau](https://nttuan8.com/bai-5-gioi-thieu-ve-xu-ly-anh/).

[Để tiện cho việc xử lý ảnh, ma trận pixel của ảnh màu thường được tách thành ba ma trận riêng biệt, mỗi ma trận biểu diễn một kênh màu (channel): kênh đỏ (red), kênh xanh lục (green), và kênh xanh lam (blue)](https://nttuan8.com/bai-5-gioi-thieu-ve-xu-ly-anh/).

[Ngoài ra, còn có các hệ màu khác như HSV (Hue, Saturation, Value) được định nghĩa theo cách tương tự như cách con người cảm nhận màu sắc](https://vi.eyewated.com/mo-hinh-mau-hsv-la-gi/).

[Độ sâu màu, hay bit màu, là thước đo số lượng màu sắc mà một bức ảnh có thể dùng để hiển thị](https://thuvien.hocviennhiepanh.com/chon-lua-8-bits-hay-16-bits-khi-hau-ky/). [Độ sâu màu cao hơn sẽ cho phép hiển thị nhiều sắc thái màu sắc hơn](https://quantrimang.com/cong-nghe/do-sau-mau-la-gi-182177).

* 1. **Màu sắc trong ảnh**
     1. **Hệ màu RGB**

RGB được viết tắt của Red (màu đỏ), Green (màu xanh lá), và Blue (màu xanh dương), là 3 màu gốc trong mô hình ánh sáng bổ sung. Các màu này tương ứng với tia sáng đỏ, xanh lá và xanh dương trong ánh sáng trắng. Mỗi màu sẽ đại diện cho 1 giá trị từ 0 – 255, trong đó, 0 là không có màu và 255 là màu sáng nhất. Sự kết hợp của 3 màu cơ bản RGB theo cường độ khác nhau cho ra các màu sắc diễn tả các màu sắc khác nhau trong tự nhiên.

Tuy nhiên, 3 màu cơ bản RGB không định nghĩa “đỏ”, “xanh lá” và “xanh dương” chính xác. Với cùng các giá trị, RGB có thể mô tả các màu khác nhau trên các thiết bị khác nhau. Trong khi cùng chia sẻ một mô hình màu chung nhưng không gian màu dao động có thể khác nhau đáng kể.

Hệ màu RGB hoạt động theo cơ chế cộng để tạo ra màu mới. Tùy vào tỷ lệ pha trộn giữa 3 màu cơ bản RGB, mỗi màu sẽ có một độ sáng khác nhau.  Khi 3 màu Red, Green và Blue hòa trộn theo tỉ lệ 1:1:1 sẽ tạo thành màu trắng trong mô hình ánh sáng bổ sung (màu sáng hơn màu gốc). Hệ thống mã hóa màu RGB (Red-Green-Blue) được sử dụng trong đồ họa máy tính sử dụng ba byte (28)3, tương đương khoảng 16 triệu màu khác nhau.

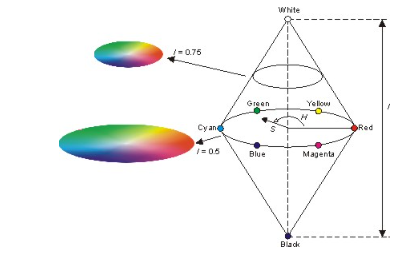
Đặc biệt là các file hệ RGB sẽ thể hiện tốt trên các thiết bị phát quang có sử dụng ánh sáng trắng làm cơ sở. Vì thế, hệ màu RGB thường được sử dụng cho các màu được thể hiện trên màn hình máy tính hoặc các màu trong lĩnh vực thiết kế digital trên website được chiếu qua các màn hình hoặc máy chiếu dùng ánh sáng.

Hiện nay, mã Hex Code 6 chữ số là phổ biến nhất. Tiếp đến là mã thập lục phân byte 3 chữ số, mỗi byte đại diện cho cường độ của màu đỏ, xanh lá và xanh dương. Giá trị byte của mã Hex nằm trong cường độ từ 00 (thấp nhất) đến FF (cao nhất)..

* + 1. **Hệ màu HSI**

**Hệ màu HSI (Hue, Saturation, Intensity)** là một mô hình màu được sử dụng rộng rãi trong xử lý ảnh số. [Hệ màu này mã hóa thông tin màu sắc theo cách mà con người thường mô tả màu sắc](https://viblo.asia/p/mo-hinh-mau-hsi-toan-tap-va-ung-dung-demo-tren-thiet-bi-di-dong-android-ZDEvLRJpeJb).

* [**Hue (H)**: Hue mang thông tin về màu sắc và có giá trị từ 0 đến 360 độ](https://viblo.asia/p/mo-hinh-mau-hsi-toan-tap-va-ung-dung-demo-tren-thiet-bi-di-dong-android-ZDEvLRJpeJb).
* **Saturation (S)**: Saturation, hay độ bão hòa, cho biết mức độ mà một màu sắc được pha loãng bằng ánh sáng trắng. [Giá trị của S nằm trong khoảng từ 0 đến 1](https://viblo.asia/p/mo-hinh-mau-hsi-toan-tap-va-ung-dung-demo-tren-thiet-bi-di-dong-android-ZDEvLRJpeJb).
* **Intensity (I)**: Intensity, hay cường độ, liên quan đến mức độ sáng của màu sắc. [Giá trị của I cũng nằm trong khoảng từ 0 đến 1](https://viblo.asia/p/mo-hinh-mau-hsi-toan-tap-va-ung-dung-demo-tren-thiet-bi-di-dong-android-ZDEvLRJpeJb).



Hệ màu HSI hoạt động dựa trên cơ sở hấp thụ ánh sáng. Màu mà chúng ta nhìn thấy là từ phần của ánh sáng không bị hấp thụ. [Để thay đổi màu trong hệ HSI, chúng ta không tăng thêm ánh sáng mà bản thân màu HSI sẽ tự loại bỏ đi ánh sáng đi từ ánh sáng gốc để thay đổi thành các màu sắc khác nhau](https://viblo.asia/p/mo-hinh-mau-hsi-toan-tap-va-ung-dung-demo-tren-thiet-bi-di-dong-android-ZDEvLRJpeJb).

**Chuyển đổi giữa không gian màu :**

**Chuyển đổi từ HSI sang RGB:**

* (H) là giá trị màu Hue nằm trong khoảng [0, 360).
* (S) là giá trị độ bão hòa Saturation nằm trong khoảng [0, 1].
* I là giá trị Intensity nằm trong khoảng [0, 1].

Tính toán hệ số R, G, B từ H, S, I theo các công thức:

Đối với các khu vực màu Hue, ánh xạ chúng vào các giá trị R, G, B như sau:

* + - Khu vực 0° ≤ (H) < 60°: (R = C, G = X, B = 0)
    - Khu vực 60° ≤ (H) < 120°: (R = X, G = C, B = 0)
    - Khu vực 120° ≤ (H) < 180°: (R = 0, G = C, B = X)
    - Khu vực 180° ≤ (H) < 240°: (R = 0, G = X, B = C)
    - Khu vực 240° ≤ (H) < 300°: (R = X, G = 0, B = C)
    - Khu vực 300° ≤ (H) < 360°: (R = C, G = 0, B = X)

Áp dụng dời và tụ điện để có giá trị cuối cùng:

* + - R = (R + m) ×255
    - G = (G + m) ×255
    - B = (B + m) ×255

**Chuyển đổi từ RGB sang HSI:**

* + (R, G, B) là giá trị màu Red, Green, Blue nằm trong khoảng [0, 255].

- Tính toán (I) (Intensity): )

- Nếu (R = G = B), có thể xác định (H) (Hue) là 0 vì ảnh đen trắng.

- Nếu (R G B), tính toán (H) từ các giá trị (R, G, B) theo các công thức:

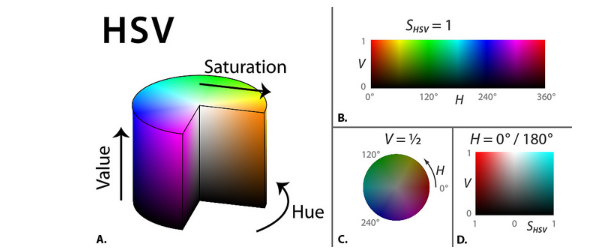
- Tính toán (S) (Saturation):

* + 1. **Hệ màu HSV**

Hệ màu HSV (Hue, Saturation, Value) là một phương pháp biểu diễn màu sắc trong không gian màu ba chiều, nơi màu sắc được mô tả bằng các yếu tố chính là màu sắc chính (Hue), độ bão hòa (Saturation), và độ sáng (Value). Hệ màu HSV thường được sử dụng trong xử lý ảnh, đồ họa máy tính và nhiều ứng dụng khác do tính linh hoạt và dễ hiểu của nó.

* + [**Hue (H)**: Hue mang thông tin về màu sắc và có giá trị từ 0 đến 360 độ](https://vi.eyewated.com/mo-hinh-mau-hsv-la-gi/).
  + **Saturation (S)**: Saturation, hay độ bão hòa, cho biết mức độ mà một màu sắc được pha loãng bằng ánh sáng trắng. [Giá trị của S nằm trong khoảng từ 0 đến 1](https://vi.eyewated.com/mo-hinh-mau-hsv-la-gi/" \t "_blank).
  + **Value (V)**: Value, hay cường độ, liên quan đến mức độ sáng của màu sắc. [Giá trị của V cũng nằm trong khoảng từ 0 đến 1](https://vi.eyewated.com/mo-hinh-mau-hsv-la-gi/" \t "_blank)

.



Màu HSV cũng cho phép họ dễ dàng thực hiện các phép toán trên màu sắc, như việc tăng hoặc giảm độ sáng, thay đổi màu sắc chính và tạo ra các hiệu ứng khác nhau. Điều này giúp cho công việc có màu sắc trở nên linh hoạt và tiện lợi.

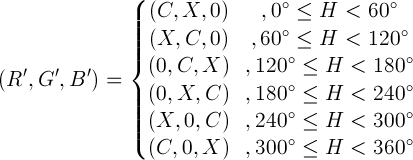
**Công thức chuyển đổi HSV sang RGB:**

Khi 0 ≤ *H* <360, 0 ≤ *S* ≤ 1 và 0 ≤ *V* ≤ 1:

*C* = *V* × *S*

*X* = *C* × (1 - | ( *H* / 60 °) mod 2 - 1 |)

*m* = *V* - *C*



( *R* , *G* , *B* ) = (( *R* '+ *m* ) × 255, ( *G* ' + *m* ) × 255, ( *B* '+ *m* ) × 255)

* + 1. **Khử nhiễu và bộ lọc nhiễu**
* **Nhiễu và mô hình nhiễu**

Ảnh được coi là một miền đồng nhất về mức xám, tức là các điểm ảnh lân cận có mức xám tương đồng. Tuy nhiên, trong quá trình số hóa ảnh, các điểm ảnh có thể bị biến đổi đột ngột và có giá trị khác biệt so với các điểm ảnh xung quanh. Đây chính là hiện tượng nhiễu trong ảnh. Nhiễu trong ảnh số xảy ra khi tín hiệu ảnh bị biến đổi đột ngột trên một khoảng cách nhỏ, tạo ra các điểm ảnh không phù hợp với ngữ cảnh xung quanh.

Có hai loại mô hình chính là mô hình liên tục và mô hình rời rạc:

* **Mô hình liên tục:**

Mô hình liên tục đại diện cho tín hiệu hoặc ảnh như một hàm liên tục trong không gian và thời gian. Trong mô hình liên tục, tín hiệu được mô tả bằng các biến thực liên tục và có thể được biểu diễn bằng các phương trình vi phân hoặc phương trình toán học khác. Mô hình liên tục cho phép xử lý và phân tích tín hiệu ở mọi mức độ chi tiết và liên tục trong thời gian và không gian.

* **Mô hình rời rạc:**

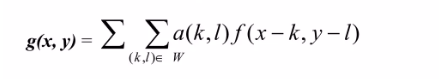
Mô hình rời rạc đại diện cho tín hiệu hoặc ảnh như một chuỗi các giá trị rời rạc theo không gian và thời gian. Trong mô hình rời rạc, tín hiệu được biểu diễn bằng các điểm dữ liệu rời rạc tại các vị trí và thời điểm cụ thể. Mô hình rời rạc thường được sử dụng trong vi xử lý ảnh và xử lý tín hiệu số, với các phép biến đổi và phép tính được thực hiện trên các giá trị rời rạc.

* + - 1. **Lọc trung bình (**Median Filter**)**

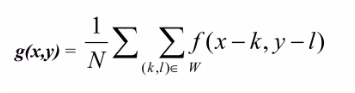
**Lọc trung bình** là sử dụng của sổ lọc của lớp tuyến tính để lọc nhiễu và làm trơn ảnh.

**Ý tưởng** : là thay thế giá trị tại mỗi điểm ảnh bằng trung bình các giá trị của điểm ảnh lân cận nhằm loại bỏ những điểm ảnh biến đổi lớn so với điểm lân cận (nhiễu), và những điểm ảnh nằm trên biên cũng có sự biến đổi lớn so với lân cận.

Lọc trung bình được biểu diễn bằng công thức toán học sau:



Nếu ta sử dụng các trọng số (cửa sổ bộ lọc), phương trình sẽ trở thành :



Trong đó:

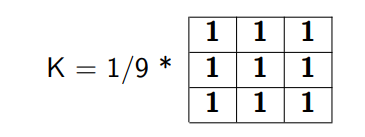
f(x,y): ảnh đầu vào

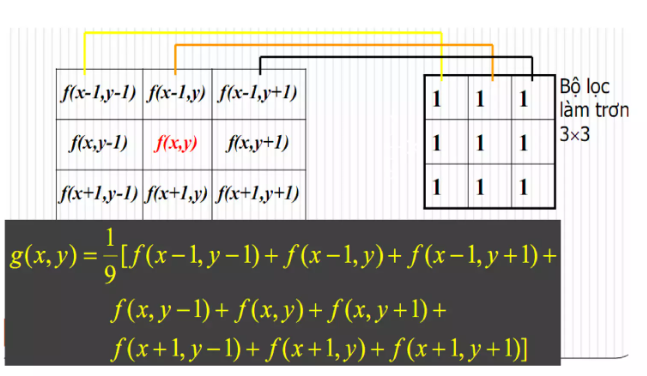
g(x,y): ảnh đầu ra

a (k,l): cửa sổ lọc( mặt lạ)

a(k,l) = và Nw  là số điểm ảnh của cửa sổ lọc W

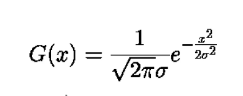
Lọc trung bình có giá trị điểm ảnh chính là tích nhân chập ảnh đầu vào với cửa sổ lọc K, Kcó dạng:

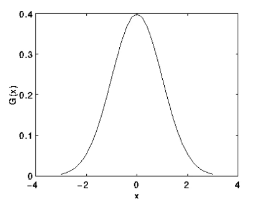




* + - 1. **Lọc Gaussian**

Lọc Gaussian là một phương pháp quan trọng trong xử lý ảnh, được sử dụng để làm mờ và giảm nhiễu hình ảnh. Nó dựa trên lý thuyết hàm Gaussian, một hàm mượt có hình dạng giống hình chuông. Công thức hàm Gaussian 1D có dạng như sau:





Trong đó :

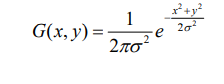
G(x) là giá trị của hàm tại điểm x.

σ (sigma) là độ lệch chuẩn (standard deviation).

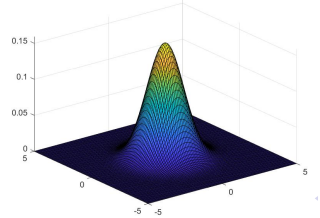
π là số pi.

(*e*) là số Euler (gần giá trị 2.71828).

Trong không gian 2 chiều, hàm Gaussian 2 chiều với trung bình (0,0) và độ lệch chuẩn bằng σ có công thức sau:



Đồ thị Gaussian có hình dạng sau:



Trong đó:

* (x, y) là tọa độ của pixel đang xét.
* sigma là tham số định rõ độ mờ của hiệu ứng. Giá trị sigma càng lớn, độ mờ càng cao.
* e là số Euler, xấp xỉ 2.71828.

Hàm Gaussian tạo ra một kernel Gaussian, có dạng hình tròn đồng tâm, với giá trị giảm dần từ trung tâm ra ngoài.

Lọc Gaussian, hay còn gọi là Gaussian Blur là một phép biến đổi sử dụng kernel Gaussian để làm mờ chi tiết và giảm nhiễu trong ảnh. Khi áp dụng Gaussian Blur, mỗi pixel trong ảnh mới được tính toán dựa trên giá trị trung bình có trọng số của các pixel xung quanh, với trọng số được xác định bởi hàm Gaussian.

Công thức áp dụng Gaussian Blur trong không gian 2 chiều:

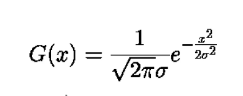
I'(x, y) =

Trong đó (I(x, y)) là giá trị pixel tại tọa độ ((x, y)) trong ảnh gốc, (I'(x, y)) là giá trị pixel sau khi áp dụng Gaussian Blur, và (G(i, j)) là giá trị của kernel Gaussian tại tọa độ ((i, j)).

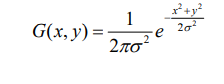
Lọc Gaussian giúp làm mờ chi tiết và giảm độ tương phản trong ảnh, tạo hiệu ứng mịn màng và làm đẹp hình ảnh. Độ lệch chuẩn (sigma) của kernel quyết định mức độ làm mờ, với (sigma) lớn tạo ra hiệu ứng làm mờ mạnh mẽ hơn.

**Chứng minh:**

Để chứng minh công thức 2D của Lọc Gaussian, ta bắt đầu bằng công thức hàm Gaussian 1D:



Chúng ta sử dụng tích phân 2D cho mỗi chiều ( x ) và ( y ) để có công thức 2D:



* Tích phân theo chiều x

=

* Tích phân theo chiều y

=

* Tích phân 2D

(x,y)dxdy=(.( )

(. ( = 2

Chia hàm G(x,y) cho tổng tích phân 2D:

G′(x,y) =

Công thức trên đảm bảo tổng tích phân 2D của nó bằng 1:

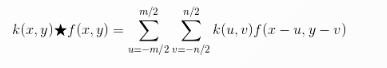
Điều này đảm bảo rằng tổng các giá trị của kernel Gaussian trên toàn bộ không gian là 1. Điều này rất quan trọng vì khi áp dụng lọc Gaussian cho một ảnh, chúng ta muốn đảm bảo tổng các pixel trong ảnh mới cũng là 1, giữ nguyên tổng giá trị ánh sáng ban đầu.

* + 1. **Cải thiện ảnh sử dụng phép nhân chập và mẫu (Convolution and Kernel)**

[Phép nhân chập (Convolution) là một kỹ thuật quan trọng trong xử lý ảnh, được sử dụng chính yếu trong các phép toán trên ảnh như đạo hàm ảnh, làm trơn ảnh, trích xuất biên cạnh trong ảnh](https://www.iostream.co/article/phep-tich-chap-trong-xu-ly-anh-convolution-r1vHu1).

[Phép nhân chập hoạt động bằng cách di chuyển bộ lọc (filter) trên khối ảnh đầu vào và tính tổng các giá trị pixel có trọng số](https://limosa.vn/convolutional-neural-network-la-gi/). [Những bộ lọc này, thường được gọi là kernel hoặc filter, được thiết kế để tìm kiếm các đặc trưng nhất định của ảnh](https://limosa.vn/convolutional-neural-network-la-gi/" \t "_blank).

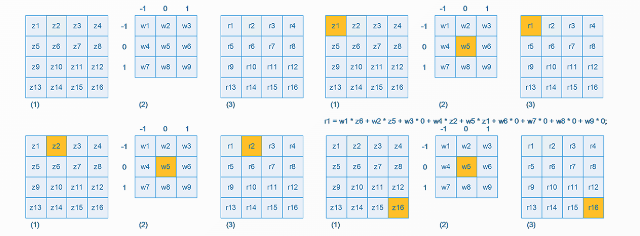
Công thức tích chập giữa hàm ảnh f(x, y) và bộ lọc k(x, y) (kích thước mxn):



Thành phần không thể thiếu của phép tích chập là ma trận kernel (bộ lọc). Điểm neo (anchor point) của kernel sẽ quyết định vùng ma trận tương ứng trên ảnh để tích chập, thông thường anchor point được chọn là **tâm** của kernel. Giá trị mỗi phần tử trên kernel được xem như là hệ số tổ hợp với lần lượt từng giá trị độ xám của điểm ảnh trong vùng tương ứng với kernel.

Phép tích chập được hình dung thực hiện bằng việc dịch chuyển ma trận kernel lần lượt qua tất cả các điểm ảnh trong ảnh, bắt đầu từ góc bên trái trên của ảnh. Và đặt anchor point tương ứng tại điểm ảnh đang xét. Ở mỗi lần dịch chuyển, thực hiện tính toán kết quả mới cho điểm ảnh đang xét bằng công thức tích chập.

Xem minh hoạ thực hiện: Ảnh minh hoạ theo thứ tự từ trái qua phải và từ trên xuống dưới. Ảnh cuối cùng là kết quả sau khi thực hiện di chuyển kernel hết toàn bộ ảnh. Ký hiệu: (1) ảnh nguồn, (2) kernel, (3) ảnh kết quả.



Một số cách xử lý vùng kernel vượt ra ngoài khỏi ảnh:

* Bỏ qua, không thực hiện tính phần tử đó vào kết quả.
* Sử dụng một hằng số để tính toán.
* Duplicate pixel nằm ở biên của ảnh.

## **Tính chất**

Tích chập được định nghĩa là 1 phép toán trên không gian khả tích của các hàm tuyến tính, cho nên nó có tính chất giao hoán, kết hợp và phân phối.

* Giao hoán: f \* g = g \* f
* Kết hợp: f \* g \* h = f \* (g \* h)
* Phân phối: f \* g + f \* h = f \* (g + h)

Do tính chất kết hợp của phép tích chập, khi một phép xử lý ảnh yêu cầu thực hiện tích chập **liên tiếp** với nhiều bộ lọc (kernel) f \* g \* h. Ta có thể tính toán trước ma trận kernel để "giảm độ phức tạp tính toán" k = v \* h do kích thước ma trận kernel hầu như rất nhỏ so với ảnh. Lúc này, thay vì thực hiện tích chập theo thứ tự r = (f \* g) \* h, ta thực hiện r = f \* (v \* h) = f \* k.

Ký hiệu:

* - f: hàm ảnh
* - g: bộ lọc thứ nhất
* - h: bộ lọc thứ hai
* - r: hàm ảnh kết quả
* **Tối ưu hàm thực hiện**

Convolution vẫn còn là một kỹ thuật với độ phức tạp tính toán cao. Một số cách dưới đây có thể tối ưu tốc độ của convolution:

* Mỗi phần tử trong ma trận kernel nên là số nguyên: như trong ví dụ trên, các phần tử trong kernel thực ra là số thực, tuy nhiên, tôi thực hiện chuyển sang ma trận số nguyên với số hạng chung cho tất cả các phần tử, kết quả tích chập sẽ nhân cho số hạng chung này.
* Kernel nên thực hiện lưu trong mảng một chiều.

Tạo ma trận chỉ số truy cập nhanh, với cách này có thể truy cập nhanh đến pixel trên ảnh, tương ứng với kernel mà không cần tính toán chỉ số thêm lần nữa.

# CHƯƠNG 2: PHƯƠNG PHÁP HIỆU CHỈNH ÁNH SÁNG

1. **Hiệu chỉnh độ sáng**

Trong ảnh kỹ thuật số, giá trị tại mỗi điểm ảnh thể hiện cường độ sáng tại điểm đó. Do đó, phương pháp đơn giản nhất để hiệu chỉnh ánh sáng trong ảnh là thay đổi giá trị điểm ảnh của ảnh. Ý tưởng của phương pháp này là thay đổi một cách đồng đều giá trị tại mỗi điểm ảnh. Phương pháp này được thực hiện bằng cách cộng giá trị mỗi điểm ảnh với một số nguyên nằm trong khoảng [-255, 255].

**Công thức điều chỉnh ánh sáng**

Công thức điều chỉnh ánh sáng được biểu diễn như sau:

Trong đó:

* g(x,y)đại diện cho giá trị cường độ điều chỉnh của pixel tại tọa độ *(x,y)* trong ảnh đầu ra.
* f(x,y)là giá trị cường độ ban đầu của pixel tại tọa độ *(x,y)* trong ảnh đầu vào.
* b là thành phần điều chỉnh độ sáng (khi *b* là giá trị âm, *g(x,y)* sẽ giảm so với *f(x,y),* đẫn đến việc làm tối ảnh , người lại *b* là giá trị dương ảnh sẽ sáng lên).

**Thuật toán điều chỉnh ánh sáng**

Dưới đây là một cách tổng quát để mô tả hoạt động của đoạn mã điều chỉnh độ sáng trong ảnh:

**Input:**

* **Ảnh gốc**: Đây là ảnh mà chúng ta muốn điều chỉnh độ sáng. Ảnh này có thể có kích thước và độ phân giải khác nhau.
* **Giá trị điều chỉnh độ sáng**: Đây là giá trị mà chúng ta muốn thêm vào mỗi điểm ảnh của ảnh gốc để tăng hoặc giảm độ sáng.

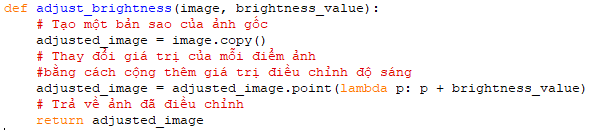
**Output:**

* **Ảnh đã điều chỉnh**: Đây là ảnh sau khi đã được điều chỉnh độ sáng. Ảnh này có cùng kích thước và độ phân giải với ảnh gốc, nhưng giá trị của mỗi điểm ảnh đã được thay đổi.

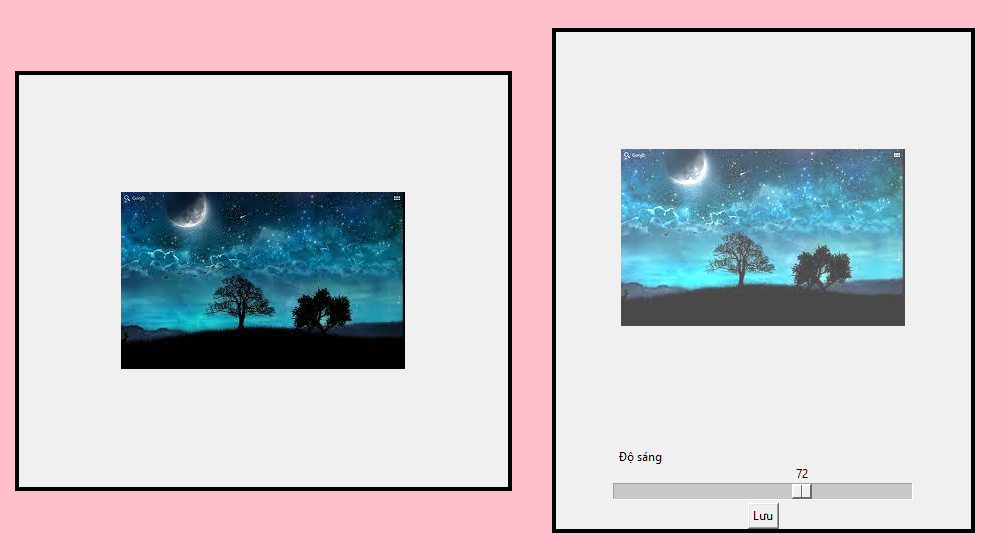
**Hoạt động của thuật toán:**

Thuật toán điều chỉnh độ sáng hoạt động bằng cách thay đổi giá trị của mỗi điểm ảnh trong ảnh gốc. Điều này được thực hiện bằng cách cộng thêm giá trị điều chỉnh độ sáng vào giá trị ban đầu của mỗi điểm ảnh.

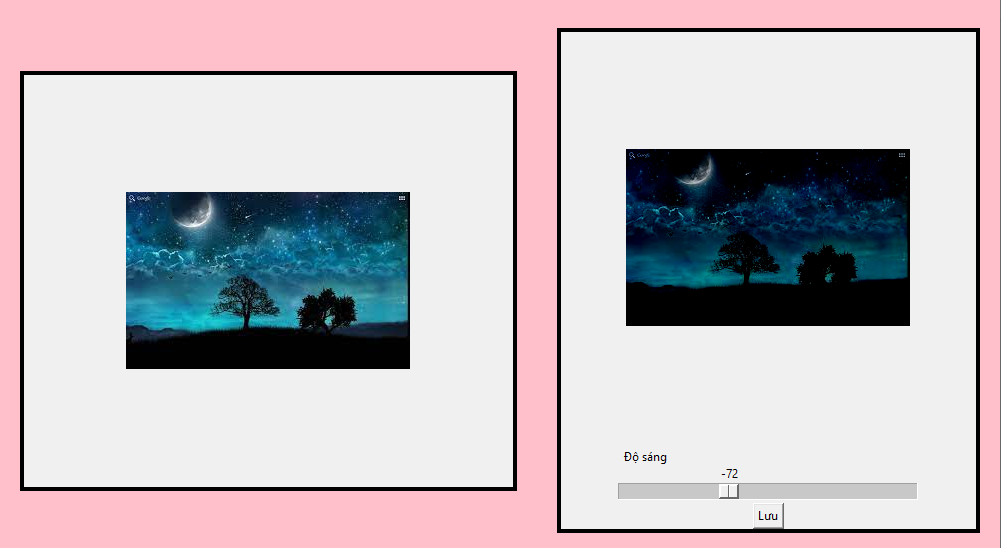
**Python**



Trong đoạn mã trên, image là ảnh gốc và brightness\_value là giá trị điều chỉnh độ sáng. Hàm ***adjust\_brightness*** sẽ trả về ảnh sau khi đã được điều chỉnh độ sáng. Kết quả cuối cùng là ảnh đã điều chỉnh, với mỗi điểm ảnh có giá trị cường độ sáng đã được thay đổi theo giá trị điều chỉnh độ sáng. Nếu brightness\_value là một số âm, ảnh sẽ tối đi. Nếu ***brightness\_value*** là một số dương, ảnh sẽ sáng lên. Ảnh gốc không bị thay đổi trong quá trình này.



*Ví dụ về chứ năng “Hiệu chỉnh ánh sáng tăng” với tham số là 72*



*Ví dụ về chứ năng “Hiệu chỉnh ánh sáng giảm” với tham số là 72*

1. **Hiệu chỉnh độ tương phản**

Độ tương phản trong ảnh đóng vai trò quan trọng trong việc xác định chất lượng hình ảnh. Nó biểu thị sự khác biệt về độ sáng giữa các điểm ảnh trong hình ảnh. Một hình ảnh với độ tương phản cao thường sắc nét hơn và dễ nhìn hơn so với hình ảnh có độ tương phản thấp.

**Điều chỉnh độ tương phản**

Điều chỉnh độ tương phản là quá trình thay đổi mức độ tương phản của hình ảnh để cải thiện chất lượng hình ảnh hoặc để phù hợp với yêu cầu cụ thể.

**Input**

* **Ảnh gốc**: Một ma trận m x n, với m là chiều rộng và n là chiều cao của ảnh. Mỗi phần tử của ma trận biểu diễn một pixel, bao gồm giá trị màu đỏ, xanh lục và xanh lam.
* **Giá trị điều chỉnh độ tương phản C**: Một số nguyên biểu thị mức độ tương phản mong muốn.

**Output**

* **Ảnh đã điều chỉnh**: Một ma trận m x n, tương tự như ảnh gốc nhưng giá trị của mỗi pixel đã được thay đổi theo công thức điều chỉnh độ tương phản.

**Công thức điều chỉnh độ tương phản**

Bước đầu tiên trong quá trình này là tính toán hệ số hiệu chỉnh độ tương phản (F), được tính theo công thức sau:

Trong đó, C biểu thị mức độ tương phản mong muốn. Để thuật toán hoạt động chính xác, giá trị cho F cần phải được lưu trữ dưới dạng số dấu phẩy động và không phải là một số nguyên.

Bước tiếp theo là thực hiện việc điều chỉnh độ tương phản thực tế. Công thức sau đây cho thấy sự điều chỉnh độ tương phản được thực hiện đối với thành phần màu đỏ của màu:

Tương tự, công thức cho thành phần màu xanh lục và xanh lam là:

Trong đó, **Truncate** là một thủ tục đảm bảo rằng các giá trị mới của màu đỏ, xanh lục và xanh lam nằm trong phạm vi hợp lệ từ 0 đến 255.Trong không gian màu RGB, giá trị màu của mỗi kênh (đỏ, xanh lục, xanh lam) nằm trong khoảng [0,255].Gía trị trung bình trong phạm vi này là 128.

Khi chúng ta trừ đi 128 từ giá trị màu gốc, chúng ta đang dịch chuyển giá trị màu sao cho giá trị trung bình (128) trở thành 0. Điều này tạo ra một phạm vi mới từ -128 đến 127.

Sau đó, chúng ta nhân giá trị màu đã được dịch chuyển này với hệ số hiệu chỉnh độ tương phản F. Điều này làm thay đổi độ tương phản của hình ảnh.

Cuối cùng, chúng ta cộng thêm 128 vào giá trị màu đã được điều chỉnh. Điều này dịch chuyển phạm vi giá trị màu trở lại phạm vi ban đầu từ 0 đến 255.

Vì vậy, giá trị 128 được sử dụng như một điểm trung tâm để dịch chuyển giá trị màu trước và sau khi điều chỉnh độ tương phản. Điều này giúp đảm bảo rằng giá trị màu sau khi điều chỉnh vẫn nằm trong phạm vi hợp lệ từ 0 đến 255.

**Phạm vi của hệ số hiệu chỉnh độ tương phản F**

Phạm vi của hệ số hiệu chỉnh độ tương phản F phụ thuộc vào giá trị của C, mức độ tương phản mong muốn :

* **Khi C = -255**: Điều này có nghĩa là bạn muốn giảm độ tương phản xuống mức tối thiểu. Khi đó, F sẽ tiến dần về 0. Khi F = 0, mọi giá trị màu sẽ trở thành 128, tức là tất cả các pixel đều có cùng mức độ sáng, làm cho hình ảnh trở nên hoàn toàn mờ.
* **Khi C = 0**: Điều này có nghĩa là bạn không muốn thay đổi độ tương phản của hình ảnh. Khi đó, F = 1, tức là hình ảnh sẽ không thay đổi.
* **Khi C = 255**: Điều này có nghĩa là bạn muốn tăng độ tương phản lên mức tối đa. Khi đó, F sẽ tiến dần về vô cùng. Khi F rất lớn, mọi giá trị màu nhỏ hơn 128 sẽ trở thành 0 và mọi giá trị màu lớn hơn 128 sẽ trở thành 255, làm cho hình ảnh trở nên rất sắc nét nhưng cũng có thể làm mất đi chi tiết.

**Thuật toán điều chỉnh độ tương phản**

Giả sử bạn có một hình ảnh I với kích thước m×n và một số nguyên b biểu thị mức độ tương phản mong muốn. Dưới đây là thuật toán để điều chỉnh độ tương phản của hình ảnh:

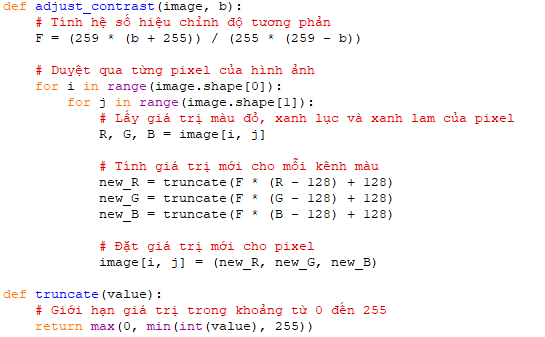
* Tính hệ số hiệu chỉnh độ tương phản F:
* Duyệt qua từng pixel (i,j) của hình ảnh I:

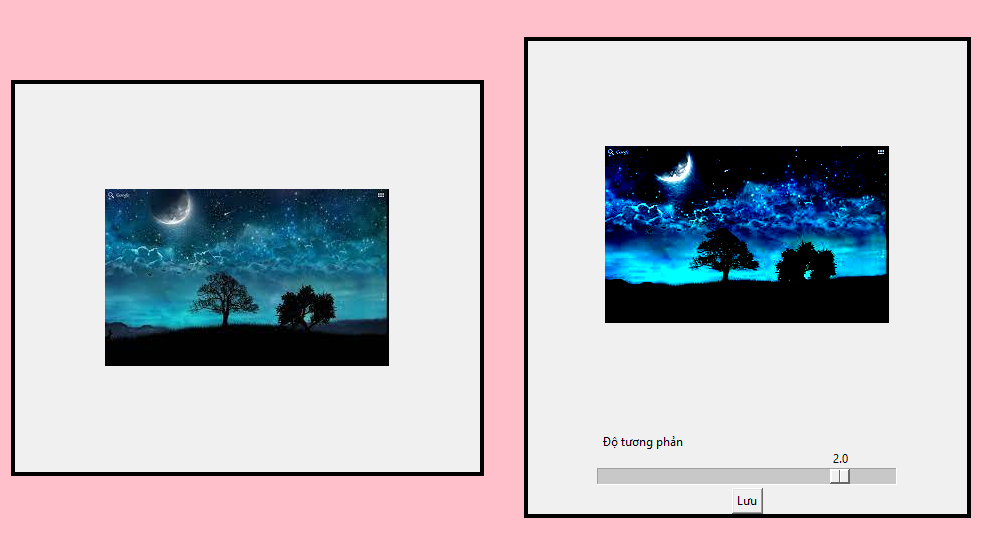
Lấy giá trị màu đỏ, xanh lục và xanh lam của pixel (i,j), ký hiệu là R, G, B.

* Tính giá trị mới cho mỗi kênh màu.
* Đặt giá trị mới cho pixel (i,j) là (**R′,G′,B′**).

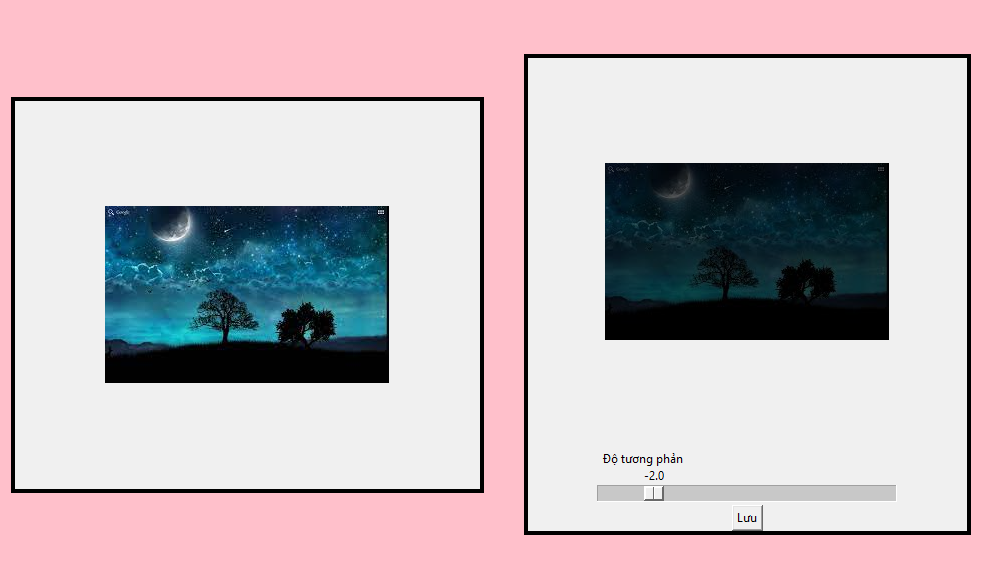
Trong đó, hàm **Truncate** được sử dụng để đảm bảo rằng giá trị mới của màu đỏ, xanh lục và xanh lam nằm trong phạm vi hợp lệ từ 0 đến 255.

Python:





*Ví dụ về chứ năng “Hiệu chỉnh tương phản” với tham số là 2.0*

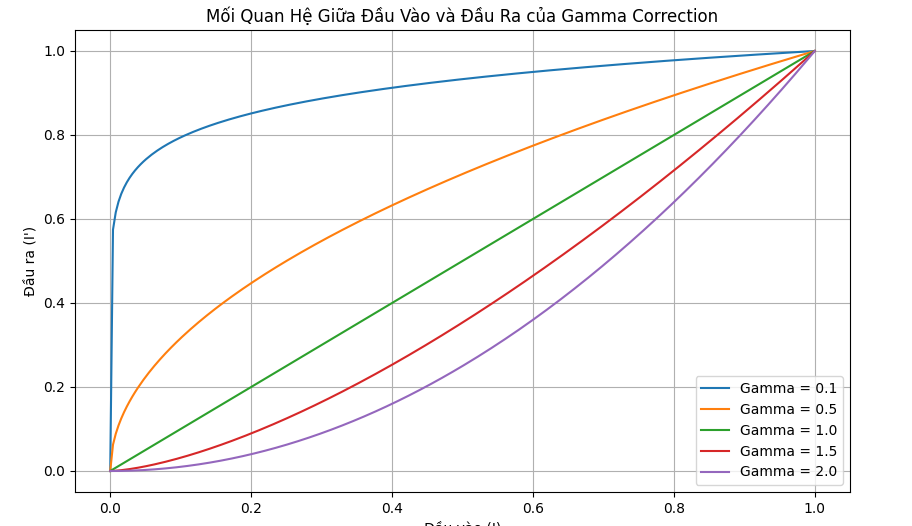


*Ví dụ về chứ năng “Hiệu chỉnh tương phản” với tham số là -2.0*

1. **Hiệu chỉnh gamma**

Khi hiển thị hình ảnh trên màn hình máy tính, một vấn đề thường gặp phải là độ nhạy sáng . Đặc điểm chung của hầu hết các loại màn hình là khi xuất kết quả, giá trị đầu ra thường là một hàm mũ của giá trị đầu vào. Điều này có thể làm giảm chất lượng hình ảnh, khiến hình ảnh hiển thị trên màn hình thường tối hơn so với bình thường.

Gamma mô tả mối quan hệ giữa giá trị đầu vào (I) và kết quả đầu ra (I’). Trong trường hợp này, đầu vào (I) là giá trị cường độ RGB của hình ảnh.



**Mối Quan Hệ Giữa Đầu Vào và Đầu Ra**

Mối quan hệ giữa đầu vào (I) và đầu ra (I’) được biểu diễn bằng công thức sau:

**I′=Iγ**

Khi gamma bằng 1, đầu vào (I) sẽ bằng đầu ra (I’), tạo ra một đường thẳng. Để thực hiện hiệu chỉnh gamma, giá trị đầu vào (I) được nâng lên lũy thừa của nghịch đảo gamma. Công thức cho việc này như sau:

**I′=I(1/γ)**

**Ảnh Hưởng Của Giá Trị Gamma**

* Khi **γ>1**, mối quan hệ giữa (I) và (I’) trở nên nhạy cảm hơn, với giá trị (I’) tăng nhanh hơn so với (I). Điều này dẫn đến việc tăng độ tương phản và làm nổi bật các chi tiết ở các mức độ sáng hoặc tối.
* Khi **0<γ<1**, mối quan hệ trở nên giảm nhạy, với giá trị (I’) tăng chậm hơn so với (I). Điều này có thể dẫn đến việc giảm độ tương phản và làm mịn các chi tiết.
* Khi **γ=1**, mối quan hệ trở thành một đường thẳng, và không có sự biến đổi ngoại trừ sự biến đổi tỷ lệ thuận tuyến tính.

**Hiệu Chỉnh Gamma**

Điều chỉnh gamma là một phương pháp xử lý hình ảnh phổ biến, được sử dụng để thay đổi độ sáng của hình ảnh. Nó hoạt động bằng cách thay đổi giá trị màu sắc của từng pixel trong hình ảnh theo một hàm mũ.

**Input:**

* **Ảnh gốc:** Một ma trận **m x n**, với **m** là chiều rộng và **n** là chiều cao của ảnh. Mỗi phần tử của ma trận biểu diễn một pixel, bao gồm giá trị màu đỏ, xanh lục và xanh lam.
* **Giá trị gamma (G):** Một số thực biểu thị mức độ điều chỉnh gamma mong muốn.

**Output:**

* **Ảnh đã điều chỉnh:** Một ma trận **m x n**, tương tự như ảnh gốc nhưng giá trị của mỗi pixel đã được thay đổi theo công thức điều chỉnh gamma.

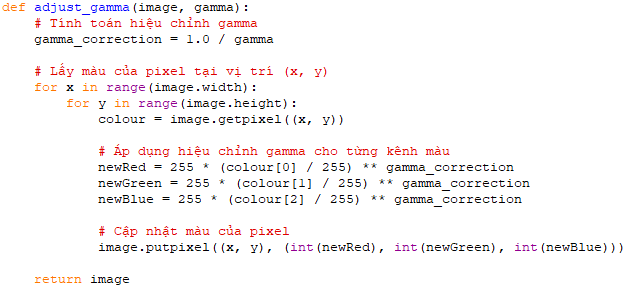
**Công thức Điều Chỉnh Gamma**

Cụ thể hơn, giá trị gammaCorrection được sử dụng trong công thức hiệu chỉnh gamma như sau:

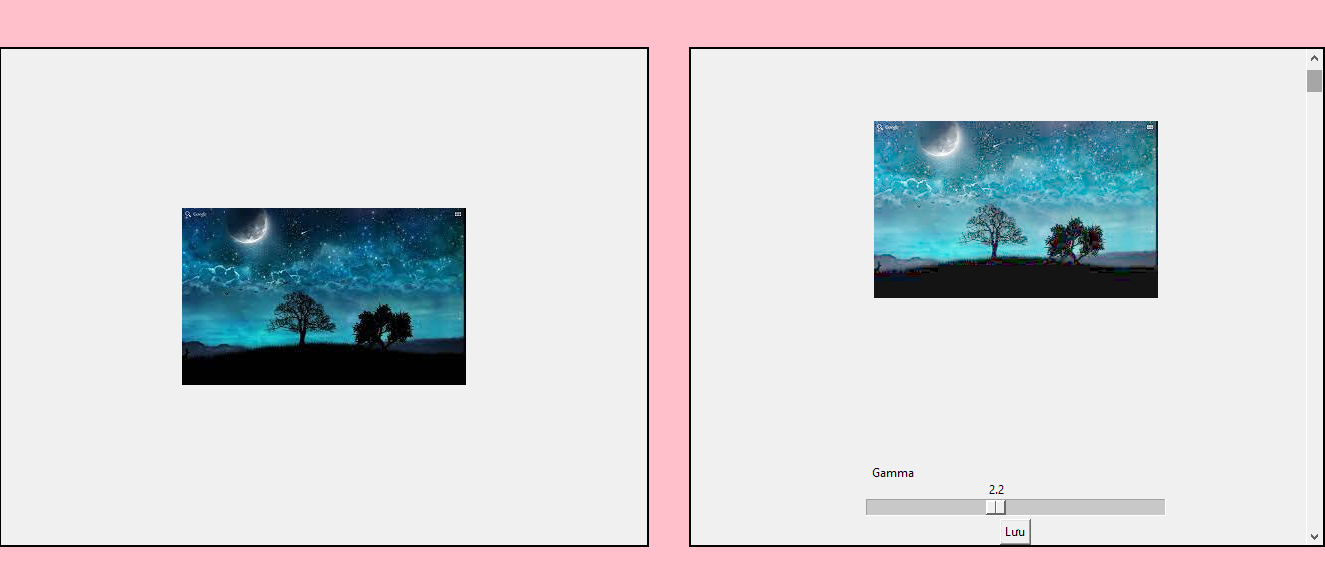
Trong đó, “I” là giá trị cường độ RGB ban đầu của một pixel (trong khoảng từ 0 đến 255), và “I’” là giá trị cường độ RGB mới sau khi đã được hiệu chỉnh gamma.

**Thuật Toán Điều Chỉnh Gamma**

Dưới đây là mã giả mô tả cách thực hiện hiệu chỉnh gamma:

****

Trong thuật toán này, chúng ta duyệt qua từng pixel của hình ảnh và áp dụng công thức điều chỉnh gamma cho mỗi kênh màu của mỗi pixel. Phạm vi giá trị được sử dụng cho gamma sẽ tùy thuộc vào ứng dụng trong xử lý ảnh.



*Ví dụ về chứ năng “Hiệu chỉnh gamma” với tham số là 2.2*

1. **Cân bằng màu**

Cân bằng màu là một kỹ thuật xử lý ảnh để cải thiện độ tương phản và sắc thái của ảnh. Có nhiều phương pháp cân bằng màu khác nhau, nhưng một trong những phương pháp phổ biến nhất là cân bằng histogram. Trong bài viết này, chúng ta sẽ tìm hiểu về histogram, cách tính và cân bằng histogram bằng Python. Chúng ta cũng sẽ thử áp dụng phương pháp này cho một ảnh mẫu và so sánh kết quả.

1. **Histogram**

Histogram là biểu đồ thống kê số lần xuất hiện của các mức sáng trong ảnh. Một ảnh có histogram cân bằng là ảnh có phân bố đều các mức sáng từ đen đến trắng, không bị chệch về một phía nào. Histogram biểu diễn độ tương phản và sắc thái của ảnh. Một ảnh có độ tương phản cao sẽ có histogram có nhiều cột cao và rộng, phân bố trên toàn dải mức sáng. Một ảnh có độ tương phản thấp sẽ có histogram có nhiều cột thấp và hẹp, tập trung ở một vùng nhỏ mức sáng. Một ảnh có sắc thái đa dạng sẽ có histogram có nhiều cột có giá trị khác nhau. Một ảnh có sắc thái đơn điệu sẽ có histogram có ít cột có giá trị gần nhau.

Cân bằng histogram là quá trình biến đổi các giá trị pixel của ảnh sao cho histogram của ảnh mới có phân bố đều hơn. Mục tiêu của cân bằng histogram là làm cho ảnh có độ tương phản cao hơn, sắc thái rực rỡ hơn và không bị chệch về một phía nào.

1. **Hàm biến đổi**

Để cân bằng histogram, ta cần xác định một hàm biến đổi **K(i)** với i là giá trị pixel ban đầu, và **K(i)** là giá trị pixel mới. Hàm **K(i)** có thể được tính theo công thức sau:

Trong đó **Z(i)** là hàm tích lũy của histogram, được tính bằng tổng số pixel có giá trị nhỏ hơn hoặc bằng **i**. **min(Z)** và **max(Z)** là giá trị nhỏ nhất và lớn nhất của **Z(i).**

Để tính toán hàm biến đổi **K(i)**, ta cần xác định hàm tính lũy **Z(i)** được tính như sau:

* Tính histogram của ảnh gốc và lưu vào mảng **H[0…L-1].**
* Khởi tạo mảng **Z[0…L-1]** với giá trị **Z(0) = H(0).**
* Duyệt qua từng mức sáng **i** từ **1** đến **L-1**: a. Tính **Z(i) = Z(i-1) + H(i).**

Hàm biến đổi **K(i)** dựa vào hàm tích lũy của histogram để chuẩn hóa các giá trị pixel về khoảng [0, 255]. Hàm biến đổi **K(i)** có tính chất là đồng biến, nghĩa là nếu **i <** **j** thì **K(i) < K(j).** Hàm biến đổi **K(i)** cũng có tính chất là bảo toàn diện tích, nghĩa là tổng số pixel của ảnh gốc và ảnh mới bằng nhau.

**Cách áp dụng hàm biến đổi**

Sau khi có hàm **K(i),** ta thay thế mỗi pixel của ảnh gốc bằng giá trị mới theo hàm **K(i).** Kết quả là ảnh mới có histogram cân bằng hơn. Cách áp dụng hàm biến đổi có thể được mô tả bằng thuật toán sau:

**Input:** Ảnh gốc **f(x, y)** có kích thước **M x N**, số mức sáng **L**.

**Output:** Ảnh mới **g(x, y)** có histogram cân bằng.

* Bước 1: Tính histogram của ảnh gốc, lưu vào mảng **H[0…L-1].**
* Bước 2: Tính hàm tích lũy của histogram, lưu vào mảng **Z[0…L-1].**
* Bước 3: Tìm giá trị nhỏ nhất và lớn nhất của hàm tích lũy, lưu vào biến **minZ** và **maxZ**.
* Bước 4: Tính hàm biến đổi **K(i)** cho mỗi mức sáng **i**, lưu vào mảng **K[0…L-1].**
* Bước 5: Duyệt qua mỗi pixel của ảnh gốc, lấy giá trị pixel là **i**, thay thế bằng giá trị mới là **K[i],** lưu vào ảnh mới.
* Bước 6: Trả về ảnh mới.

1. **Chọn không gian màu**

Cân bằng histogram có thể được áp dụng cho ảnh xám hoặc ảnh màu.Tuy nhiên, với ảnh màu , việc chọn không gian màu đúng là quan trọng để tránh thay đổi màu sắc không mong muốn. Một số không gian màu phổ biến như RGB,CMYK, HSV,HSL,…

* **Ảnh xám**

Đối với ảnh xám, quá trình này được thực hiện trên một kênh duy nhất đại diện cho độ sáng.

Trước tiên, chúng ta chuyển đổi ảnh màu sang ảnh xám để làm giảm chiều sâu màu sắc, chỉ giữ lại thông tin độ sáng.



Sau khi có ảnh xám, chúng ta sử dụng hàm cv2.equalizeHist() để áp dụng cân bằng histogram.





*Hình ảnh trước và sau khi cân bằng màu với ảnh xám*

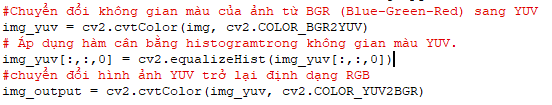
* **Ảnh màu**

Với ảnh màu, không nên cân bằng histogram trên không gian màu RGB trực tiếp. Bởi vì RGB biểu diễn màu sắc bằng ba kênh đỏ, lục và lam, mỗi kênh có ảnh hưởng đến cả độ sáng và độ bão hòa của ảnh. Nếu cân bằng histogram trên từng kênh RGB riêng biệt, màu sắc của ảnh có thể thay đổi không mong muốn.

Mặc định ảnh màu là hệ RGB hoặc BGR, muốn cân bằng sáng ta cần biến đổi về hệ màu HSV. Hệ màu HSV bao gồm 3 chanel:

* H-HUE: giá trị màu
* S-SATURATION: độ bảo hòa.
* V- VALUE: độ sáng của màu sắc.

Ta sẽ áp dụng cân bằng histogram chỉ trên độ sáng V của ảnh:





*Hình ảnh trước và sau khi cân bằng màu với ảnh màu*

# CHƯƠNG 3: THỰC NGHIỆM