TRƯỜNG ĐẠI HỌC DUY TÂN

KHOA SAU ĐẠI HỌC

Tiểu luận môn:

XỬ LÝ ẢNH

Đề tài : Hiệu chỉnh màu và ánh sáng trong ảnh số

Giảng viên : PGS.TS Đỗ Năng Toàn

Học viên : Phạm Minh Tuấn

Lớp : Khóa 22, ngành Khoa học máy tính

Đà nẵng, 12/2020

LỜI MỞ ĐẦU

Trong xã hội hiện nay, ảnh số đóng một vai trò quan trọng và hiện diện ở hầu hết các lĩnh vực của xã hội. Với tầm quan trọng và mức độ phổ biến, ảnh số được sử dụng như một phương tiện cung cấp cung cấp thông tin về vật thể, sự kiện cũng như các lĩnh vực khác trong nghiên cứu khoa học.

Trong một bức ảnh, ánh sáng cũng như màu sắc đóng vai trò rất quan trọng, nó ảnh hưởng trực tiếp lên chất lượng của bức ảnh. Vì vậy, để khắc phục các điểm yếu của trang thiết bị thu nhận cũng như điều kiện ngoại cảnh thì việc áp dụng các thuật toán xử lý để nâng cao chất lượng hình ảnh đầu vào là rất cần thiết. Trong nội dung tiểu luận, một vài phương pháp cơ bản của xử lý màu và ánh sáng được hiện thực hóa qua nội dung được trình bày bên dưới:

Chương 1: Khái niệm về xử lý ảnh số và hiệu chỉnh ánh sáng và màu trong ảnh số.

Chương 2: Một số phương pháp và thuật toán hiệu chỉnh ánh sáng và màu trong ảnh số.

Chương 3: Giới thiệu chương trình thực thi và các chức năng của chương trình.

MỤC LỤC

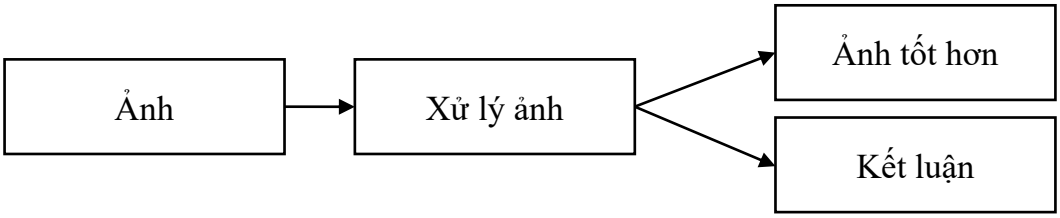
CHƯƠNG 1: KHÁI QUÁT VỀ XỬ LÝ ẢNH VÀ HIỆU CHỈNH MÀU SẮC

I. KHÁI QUÁT VỀ XỬ LÝ ẢNH

1. Xử lý ảnh là gì?

Con người thu nhận thông tin qua các giác quan, trong đó thị giác đóng vai trò quan trọng nhất. Những năm trở lại đây với sự phát triển của phần cứng máy tính, xử lý ảnh và đồ hoạ đó phát triển một cách mạnh mẽ và có nhiều ứng dụng trong cuộc sống. Xử lý ảnh và đồ họa đóng một vai trò quan trọng trong tương tác giữa người và máy tính.

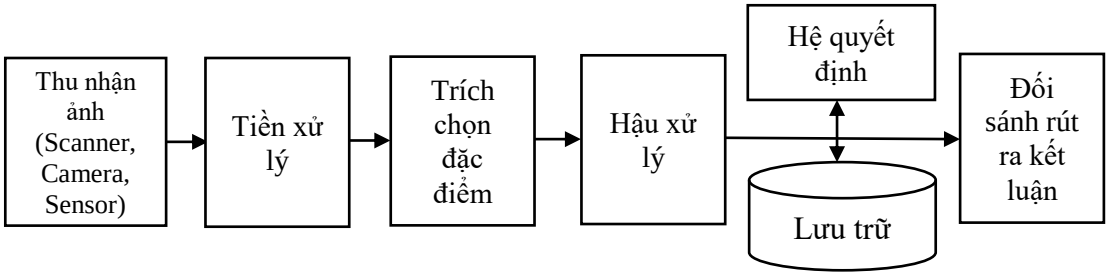
Quá trình xử lý ảnh được xem như là quá trình thao tác ảnh đầu vào nhằm đưa ra kết quả mong muốn. Kết quả đầu ra của một quá trình xử lý ảnh có thể là một ảnh "tốt hơn" hoặc một kết luận.



Hình 1.1 - Quá trình xử lý ảnh

Ảnh có thể xem là tập hợp các điểm ảnh và mỗi điểm ảnh được xem như là đặc trưng cường độ sáng hay một dấu hiệu nào đó tại một vị trí nào đó của đối tượng trong không gian và nó có thể xem như một hàm n biến P(c1, c2,..., cn). Do đó, ảnh trong xử lý ảnh có thể xem như ảnh n chiều.

Sơ đồ tổng quát của một hệ thống xử lý ảnh:



Hình 1.2 – Các bước cơ bản trong một hệ thống xử lý ảnh

2. Một số khái niệm cơ bản trong xử lý ảnh

- **Điểm ảnh và ảnh**

Gốc của ảnh (ảnh tự nhiên) là ảnh liên tục về không gian và độ sáng. Để xử lý bằng máy tính (số), ảnh cần phải được số hóa. Số hóa ảnh là sự biến đổi gần đúng một ảnh liên tục thành một tập điểm phù hợp với ảnh thật về vị trí (không gian) và độ sáng (mức xám). Khoảng cách giữa các điểm ảnh đó được thiết lập sao cho mắt người không phân biệt được ranh giới giữa chúng. Mỗi một điểm như vậy gọi là điểm ảnh (pixel).

- **Độ phân giải của ảnh (Resolution)**

Độ phân giải là mật độ điểm ảnh được ấn định trên một ảnh số được hiển thị. Theo định nghĩa, khoảng cách giữa các điểm ảnh phải được chọn sao cho mắt người vẫn thấy được sự lien tục của ảnh. Việc lựa chọn khoảng cách thích hợp tạo nên một mật độ phân bổ, đó chính là độ phân giải và được phân bố theo trục x và y trong không gian hai chiều.

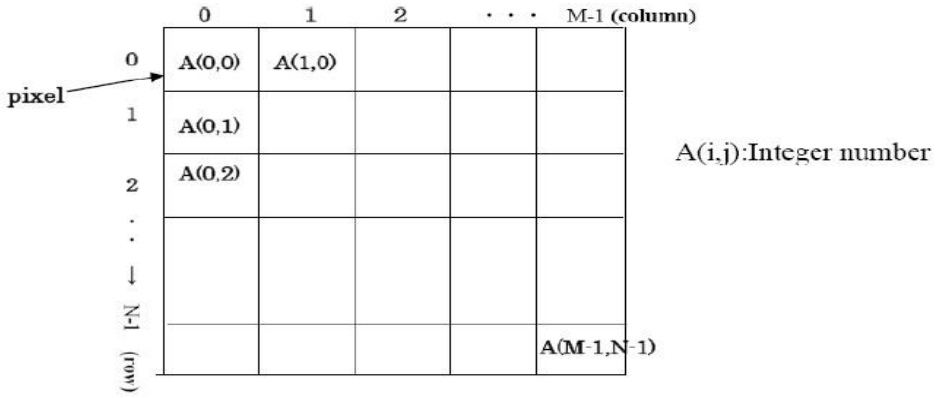
- **Mức xám của ảnh (Gray level)**

Là kết quả của sự biến đổi tương ứng một giá trị độ sáng của một điểm ảnh với một giá trị nguyên dương. Thông thường nó xác định trong khoảng 0…255. Tùy thuộc vào giá trị xám mà mỗi điểm ảnh được biểu diễn trên 1, 4, 8, 24 hay 32 bit.

3. Cấu trúc của ảnh số

Là tập hợp hữu hạn các điểm ảnh. Ảnh có thể được biểu diễn dưới dạng một ma trận 2 chiều, mỗi phần tử của ma trận tương ứng với một điểm ảnh. Mỗi phần tử này được gọi là một pixel (picture element).

Ảnh có thể được định nghĩa là một hàm 2 chiều f(x, y), trong đó x và y là các tọa độ trong không gian (spatial) hoặc mặt phẳng (plane), và độ lớn (amplitude) của hàm f được gọi là độ sáng (intensity) hay độ xám (gray level) của ảnh tại điểm đó. Ảnh số chỉ là một ma trận 2 chiều, và việc xử lý chúng chỉ là những thao tác trên ma trận này sao cho ra kết quả hợp lý.



Hình 1.3. Biểu diễn ảnh số bằng ma trận 2 chiều

4. Phân loại ảnh số :

Tùy theo giá trị dùng để biểu diễn điểm ảnh mà người ta phân ra 3 loại ảnh chính:

- **Ảnh nhị phân (binary image)**

Giá trị mỗi điểm ảnh là 0 hoặc 1, nghĩa là trắng hoặc đen. Trong thực tế khi xử lý trên máy tính thì người ta dùng ảnh xám (xem khái niệm bên dưới) để biểu diễn ảnh nhị phân và lúc này 2 giá trị là 0 hoặc 255.

- **Ảnh xám (Grayscale)**

Ảnh xám hay còn gọi là ảnh đơn sắc (monochromatic), là ảnh mà tại mỗi điểm ảnh có một giá trị mức xám. Ảnh 8 mức xám sẽ có giá trị mỗi điểm ảnh nằm trong đoạn [0, 7]. Ảnh 256 mức xám sẽ có giá trị mỗi điểm ảnh nằm trong đoạn [0, 255].

Giá trị điểm ảnh = 0 nghĩa là điểm ảnh đó tối (đen), giá trị điểm ảnh lớn nhất nghĩa là điểm ảnh đó trắng. Nói cách khác, giá trị mỗi điểm ảnh càng lớn thì điểm ảnh đó càng sáng. Cường độ sáng được tính theo công thức (chuyển đổi từ RGB sang grayscale): độ sáng = 0.2989R + 0.5870G + 01.1140B. Các hệ số có thể được làm tròn thành 0.3, 0.59 và 0.11.

- **Ảnh màu (Color image)**

Ảnh màu là ảnh mà tại mỗi điểm là một cấu trúc gồm nhiều kênh màu khác nhau (thường là 3 kênh màu). Cũng như ảnh xám, mỗi kênh màu trong ảnh màu có thể được mã hoá n bit cho mỗi kênh. Như vậy 1 pixel sẽ được mã hoá bởi 3*n bit. Do đó dung lượng ảnh màu sẽ lớn hơn 3 lần so với ảnh xám có cùng mức mã hoá.

II. HIỆU CHỈNH ÁNH SÁNG VÀ MÀU SẮC

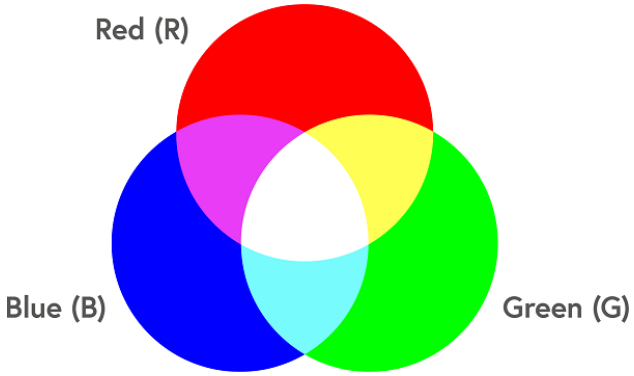
1. Ánh sáng và màu sắc

Như đã giới thiệu ở trên, hình ảnh được số hóa dưới dạng ma trận điểm ảnh. Điểm ảnh được xem như là dấu hiệu hay cường độ sáng tại 1 toạ độ trong không gian của đối tượng. Giá trị có thể có của mỗi điểm ảnh là mức xám, màu sắc của điểm ảnh đó. Và do đó ánh sáng trong ảnh số cũng chính là màu sắc trong ảnh.

2. Một số hệ màu

- **Hệ màu RGB**

Mắt người có thể phân biệt hàng ngàn màu sắc khác nhau, những con số chính xác hơn vẫn còn đang được bàn cãi nhiều. Ba màu RGB (Red-Green- Blue) mã hóa hệ thống đồ họa sử dụng ba byte (28)3 hay khoảng chừng 16 triệu màu phân biệt. Máy tính có thể phân biệt bất kỳ màu gì sau khi được mã hóa, nhưng việc mã hóa có thể không trình bày được những sự khác biệt trong thế giới thực. Mỗi điểm ảnh RGB bao gồm một byte cho màu R, một byte cho màu G và một byte cho màu B.



Việc mã hóa một màu tùy ý trong dãy hiển thị được làm bằng cách tổ hợp ba màu chính. Ví dụ: Red(255,0,0), Green(0,255,0), Blue(0,0,255), Black(0,0,0) Hệ thống màu RGB là một hệ thống màu cộng vào bởi vì mỗi màu được tạo nên bằng cách cộng thêm các phần tử vào màu đen(0,0,0)

Khuôn dạng của không gian màu RGB là định dạng phổ biến nhất của ảnh số, lý do chính là tính tương thích với màn hình hiển thị chính là màn hình vi tính. Tuy nhiên không gian màu RGB có hạn chế lớn nhất là không phù hợp với cách con người cảm nhận về màu sắc. Do đó không phù hợp cho việc ứng dụng vào tìm kiếm ảnh.

- **Hệ màu CMY và CMYK**

CMYK là từ viết tắt tiếng Anh của cơ chế hệ màu trừ, thường được sử dụng trong in ấn. Nó bao gồm các màu sau:

C = Cyan (xanh)

M = Magenta (hồng)

Y = Yellow (vàng)

K = Black (Đen) (sở dĩ dùng từ K để chỉ màu đen vì ký tự B đã được dùng để chỉ màu Blue, ngoài ra K còn có nghĩa là Key, mang ý chỉ cái gì đó là chủ yếu, là then chốt)

Nguyên lý làm việc chính của hệ CMYK là hấp thụ ánh sáng. Màu mà ta nhìn thấy là từ phần của ánh sáng không bị hấp thụ, hay nói cách khác, chúng hoạt động trên cơ chế những vật không tự phát ra ánh sáng mà chỉ phản xạ ánh sáng từ các nguồn khác chiếu tới.

Do đó thay vì thêm độ sáng để có những màu sắc khác nhau, CMYK sẽ loại trừ ánh sáng đi từ ánh sáng gốc là màu trắng để tạo ra các màu sắc khác. 3 màu Cyan, Magenta và Yellow khi kết hợp sẽ tạo ra một màu đen.

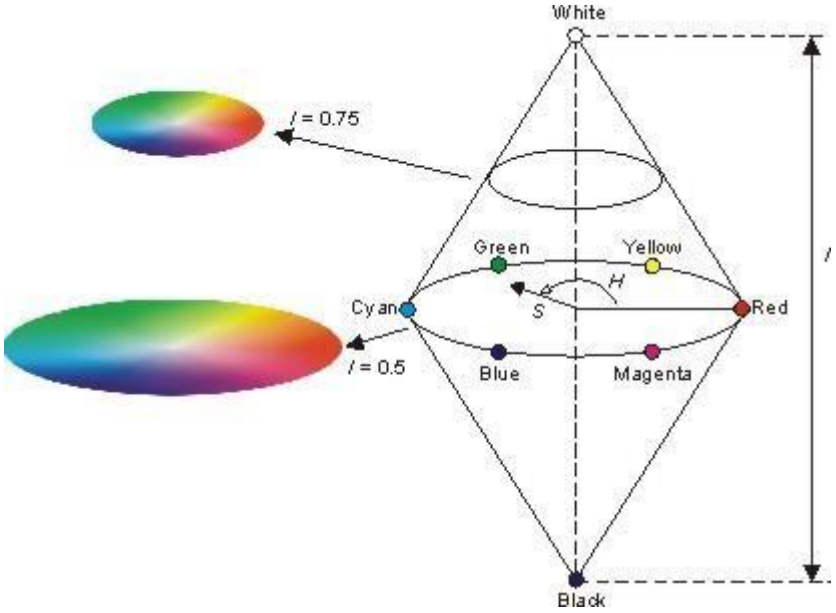
Màu CMYK thường được sử dụng khi thiết kế phục vụ cho mục đích in ấn các thiết kế như poster, brochure, name card, catalogue, sách hoặc tạp chí,…

- **Hệ màu HSI**

Hệ thống màu HSI mã hóa thông tin màu sắc bằng cách chia giá trị intensity(I) từ hai giá trị được mã hóa thuộc về độ hội tụ của màu - hue(H) và saturation(S).

Thành phần không gian màu HSI gồm có ba phần: Hue được định nghĩa có giá trị 0-2Π , mang thông tin về màu sắc. Saturation có giá trị 0-1, mang giá trị về độ thuần khiết của thành phần Hue. Intensity (Value) mang thông tin về độ sáng của điểm ảnh.Ta có thể hình dung không gian màu HSI như là vật hình nón. Với trục

chính biểu thị cường độ sáng Intensity. Khoảng cách đến trục biểu thị độ tập trung Saturation. Góc xung quanh trục biểu thị cho sắc màu Hue.



Hệ thống màu HSI thì thích hợp hơn với một số thiết kế đồ họa bởi vì nó cung cấp sự điều khiển trực tiếp đến ánh sáng và hue. Hệ thống màu HSI cũng hỗ trợ tốt hơn cho những thuật toán xử lý ảnh vì sự tiêu chuẩn hóa về ánh sáng và tập chung vào hai tham số về độ hội tụ màu, và cường độ màu.

3. Hiệu chỉnh ánh sáng trong ảnh số

Hiệu chỉnh ánh sáng trong ảnh là một kỹ thuật máy tính nhằm xử lý ánh sáng trong ảnh nhằm đạt một mục đích nào đó của người dùng. Ánh sáng có thể được làm tăng, giảm hoặc loại bỏ tác động của nó lên bức ảnh.

Hiệu chỉnh ánh sáng trong ảnh là một bước quan trọng trong hệ thống xử lý ảnh. Thực hiện hiệu chỉnh ánh sáng tốt có thể tạo ra những lợi thế rất lớn cho các công việc xử lý ảnh khác sau này.

Ảnh chụp cùng một cảnh có thể thay đổi rất nhiều bởi điều kiện ánh sáng và góc chụp của camera. Chẳng hạn như chụp thẳng, chụp nghiêng, chụp ngược sáng… Với điều kiện ánh sáng khác nhau, một số chi tiết trong ảnh sẽ bị biến đổi màu sắc hoặc thậm chí bị mờ.

- Nguồn sáng làm thay đổi màu sắc: ánh sáng chiếu vào một vật thể làm mắt người nhận thấy một phần vật thể hoặc toàn vật thể có màu sắc khác.

- Ánh sáng quá chói hoặc quá tối: ảnh chụp dưới điều kiện ánh sáng quá sáng hoặc quá tối có thể gây ra ảnh bị mờ, nhiễu, mất chi tiết.

CHƯƠNG 2: MỘT SỐ PHƯƠNG PHÁP HIỆU CHỈNH MÀU SẮC

1. Chuyển ảnh sang màu âm bản (Invert)

Để tạo ảnh âm bản, ta dùng hàm đảo ngược độ sáng (intensity) của ảnh gốc. Cách xử lí này nhằm tăng cường các chi tiết trắng hoặc xám nằm trong các vùng tối của ảnh, nhất là khi các vùng đen chiếm diện tích nhiều hơn.

Giải pháp này thực hiện bằng cách ta lấy 255 trừ đi giá trị màu tại mỗi điểm ảnh.

Giả sử ta có ảnh I với kích thước m × n và một số nguyên c. Khi đó thuật toán được mô tả như sau:

for (i = 0; i < m; i + +) {

for (j = 0; j < n; j + +) {

I [i, j] = 255 - I [i, j];

}

}

Hàm xử lý được viết bằng ngôn ngữ C#:

```
public static Bitmap Invert(Bitmap sourceBitmap)
{
    Bitmap invertBitmap = sourceBitmap.Clone() as Bitmap;
    Color color;
    for (int i = 0; i < invertBitmap.Width; i++)
    {
        for (int j = 0; j < invertBitmap.Height; j++)
        {
            color = invertBitmap.GetPixel(i, j);
            invertBitmap.SetPixel(i, j, Color.FromArgb(255 - color.R, 255 - color.G, 255 - color.B));
        }
    }
    return invertBitmap;
}
```

2. Hiệu chỉnh áng sáng (Brightness)

Giá trị tại mỗi điểm ảnh thể hiện cường độ sáng tại đó. Vì vậy để hiệu chỉnh ánh sáng ta cần thay đổi giá trị điểm ảnh.

Phương pháp này thực hiện bằng cách cộng giá trị của mỗi điểm ảnh với một số nguyên nằm trong khoảng [-255, 255].

Giả sử ta có ảnh I với kích thước m × n và một số nguyên c. Khi đó thuật toán được mô tả như sau:

for (i = 0; i < m; i + +) {

for (j = 0; j < n; j + +) {

I [i, j] = I [i, j] + c;

}

}

Nếu c > 0: ảnh sáng lên.

Nếu c < 0: ảnh tối đi.

Hàm xử lý được viết bằng ngôn ngữ C#:

```
public static Bitmap Brightness(Bitmap sourceBitmap, int brightness)
{
    Bitmap brightnessBitmap = sourceBitmap.Clone() as Bitmap;
    Color color;
    for (int i = 0; i < brightnessBitmap.Width; i++)
    {
        for (int j = 0; j < brightnessBitmap.Height; j++)
        {
            color = brightnessBitmap.GetPixel(i, j);
            int cR = color.R + brightness;
            int cG = color.G + brightness;
            int cB = color.B + brightness;

            if (cR < 0) cR = 1;
            if (cR > 255) cR = 255;

            if (cG < 0) cG = 1;
            if (cG > 255) cG = 255;

            if (cB < 0) cB = 1;
            if (cB > 255) cB = 255;

            brightnessBitmap.SetPixel(i, j, Color.FromArgb((byte)cR, (byte)cG, (byte)cB));
        }
    }
    return brightnessBitmap;
}
```

3. Hiệu chỉnh độ tương phản (Contrast)

Mục đích của thuật toán này là tạo sự thay đổi một cách rõ ràng giữa các điểm ảnh để sau khi thực hiện ta thu được một ảnh với các đối tượng được phân biệt rõ ràng hơn.

Phương pháp này được thực hiện bằng cách nhân giá trị mỗi điểm ảnh với một số nguyên dương.

Giả sử ta có ảnh I với kích thước m × n và một số nguyên c. Khi đó thuật toán được mô tả như sau:

for (i = 0; i < m; i + +) {

for (j = 0; j < n; j + +) {

I [i, j] = I [i, j] x c;

}

}

Nếu c > 1: tăng độ tương phản.

Nếu c < 1: giảm độ tương phản.

Hàm xử lý được viết bằng ngôn ngữ C#:

```
public static Bitmap Contrast(Bitmap sourceBitmap, double contrast)
{
    Bitmap contrastBitmap = sourceBitmap.Clone() as Bitmap;

    contrast = (100.0 + contrast) / 100.0;
    contrast *= contrast;

    Color color;
    for (int i = 0; i < contrastBitmap.Width; i++)
    {
        for (int j = 0; j < contrastBitmap.Height; j++)
        {
            color = contrastBitmap.GetPixel(i, j);

            double pR = color.R / 255.0;
            pR -= 0.5;
            pR *= contrast;
            pR += 0.5;
            pR *= 255;
            if (pR < 0) pR = 0;
            if (pR > 255) pR = 255;

            double pG = color.G / 255.0;
            pG -= 0.5;
            pG *= contrast;
            pG += 0.5;
            pG *= 255;
            if (pG < 0) pG = 0;
            if (pG > 255) pG = 255;

            double pB = color.B / 255.0;
            pB -= 0.5;
            pB *= contrast;
            pB += 0.5;
            pB *= 255;
            if (pB < 0) pB = 0;
            if (pB > 255) pB = 255;

            contrastBitmap.SetPixel(i, j, Color.FromArgb((byte)pR, (byte)pG, (byte)pB));
        }
    }
    return contrastBitmap;
}
```

4. Chuyển ảnh sang ảnh đa mức xám (GrayScale)

Cường độ sáng được tính theo công thức (chuyển đổi từ RGB sang grayscale): độ sáng = 0.2989R + 0.5870G + 01.1140B. Các hệ số có thể được làm tròn thành 0.3, 0.59 và 0.11.

Giả sử ta có ảnh I với kích thước m × n. Khi đó thuật toán được mô tả như sau:

for (i = 0; i < m; i + +) {

for (j = 0; j < n; j + +) {

I [i, j] = 0.2989R(i,j) + 0.5870G(i,j) + 01.1140B(i,j);

}

}

Hàm xử lý được viết bằng ngôn ngữ C#:

```
public static Bitmap GrayScale(Bitmap sourceBitmap)
{
    Bitmap grayscaleBitmap = sourceBitmap.Clone() as Bitmap;
    Color color;
    for (int i = 0; i < grayscaleBitmap.Width; i++)
    {
        for (int j = 0; j < grayscaleBitmap.Height; j++)
        {
            color = grayscaleBitmap.GetPixel(i, j);
            byte gray = (byte)(.299 * color.R + .587 * color.G + .114 * color.B);

            grayscaleBitmap.SetPixel(i, j, Color.FromArgb(gray, gray, gray));
        }
    }
    return grayscaleBitmap;
}
```

5. Hiệu chỉnh Gamma

Bản chất của việc hiển thị máy tính là việc đưa hình ảnh dạng dữ liệu ra màn hình (output). Tuy nhiên trong quá trình hiển thị trên màn hình, thiết bị đầu cuối thường gặp 1 vấn đề là độ nhạy sáng (Light Intensity). Hầu như các loại màn hình đều có đặc điểm chung là khi xuất kết quả đều cho ra giá trị là một hàm mũ. Tức là với x là giá trị đầu vào thì khi xuất ra màn hình sẽ là lũy thừa của x. Điều này này làm giảm chất lượng hình ảnh và hình ảnh thường tối hơn.

Do đó, để hình ảnh có độ hiển thị trung thực, ảnh đầu vào sẽ được làm lũy thừa với một số mũ gọi là gamma.

Giả sử màn hình đưa ra kết quả là lũy thừa với số mũ là 2.5 thì ảnh đầu vào khi thực hiện "hiệu chỉnh gamma" sẽ được làm lũy thừa với số mũ $\gamma = 1/2.5$.

Giả sử ta có ảnh I với kích thước m × n và một số nguyên c. Khi đó thuật toán được mô tả như sau:

```
for (i = 0; i < m; i + +) {
        for (j = 0; j < n; j + +)  {
                I [i, j] = Round(255*Pow(I [i, j]/255, gamma)+0.5);
        }
}
```

Nếu gamma < 1: Ảnh sáng lên

Nếu gamma > 1: Ảnh tối đi

Hàm xử lý được viết bằng ngôn ngữ C#:

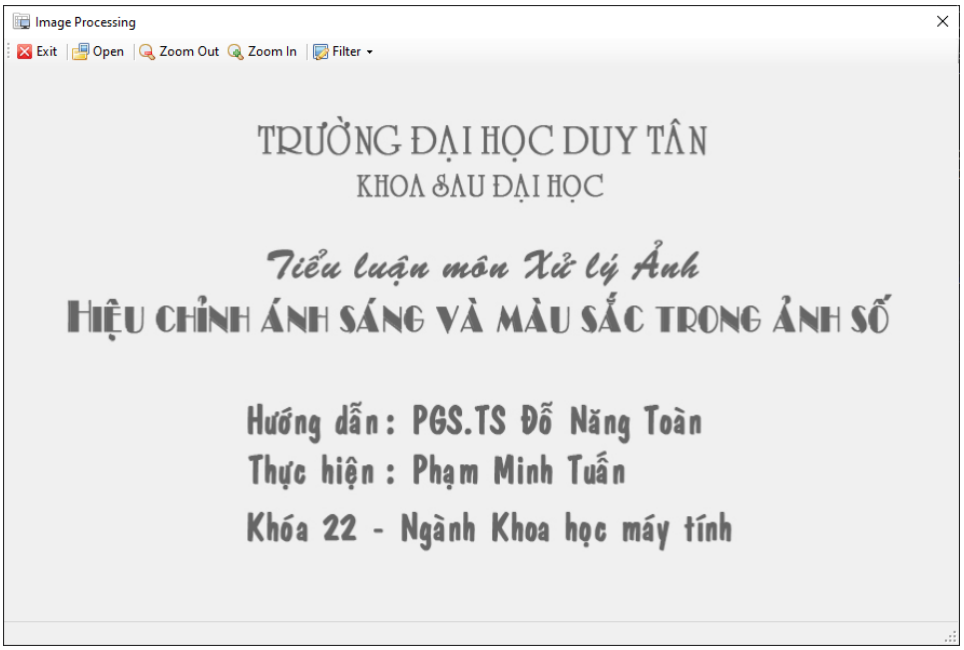
```
public static Bitmap Gamma(Bitmap sourceBitmap, double red, double green, double blue)
{
    byte[] redGamma = new byte[256];
    byte[] greenGamma = new byte[256];
    byte[] blueGamma = new byte[256];

    for (int i = 0; i < 256; ++i)
    {
        redGamma[i] = (byte)Math.Min(255, (int)((255.0 * Math.Pow(i / 255.0, 1.0 / red)) + 0.5));
        greenGamma[i] = (byte)Math.Min(255, (int)((255.0 * Math.Pow(i / 255.0, 1.0 / green)) + 0.5));
        blueGamma[i] = (byte)Math.Min(255, (int)((255.0 * Math.Pow(i / 255.0, 1.0 / blue)) + 0.5));
    }

    Bitmap gammaBitmap = sourceBitmap.Clone() as Bitmap;
    Color color;

    for (int i = 0; i < gammaBitmap.Width; i++)
    {
        for (int j = 0; j < gammaBitmap.Height; j++)
        {
            color = gammaBitmap.GetPixel(i, j);
            gammaBitmap.SetPixel(i, j, Color.FromArgb(redGamma[color.R], greenGamma[color.G], blueGamma[color.B]));
        }
    }
    return gammaBitmap;
}
```

CHƯƠNG 3: CHƯƠNG TRÌNH THỰC TẾ

1. Giới thiệu chương trình

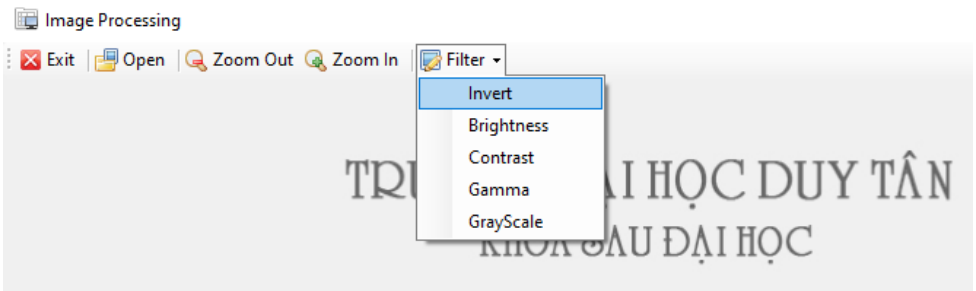
Chương trình "Xử lý ảnh (ImageProcessing)" sử dụng ngôn ngữ C# để phát triển và các thủ thuật khác nhằm minh họa cho các thuật toán trình bày trong tiểu luận.

Chương trình thiết kế trên .Net Framework 2.0 của Microsoft nên có thể chạy trên tất cả các hệ điều hành Windows.



Hình 3.1.1 – Giao diện chính của chương trình

2. Chức năng chương trình

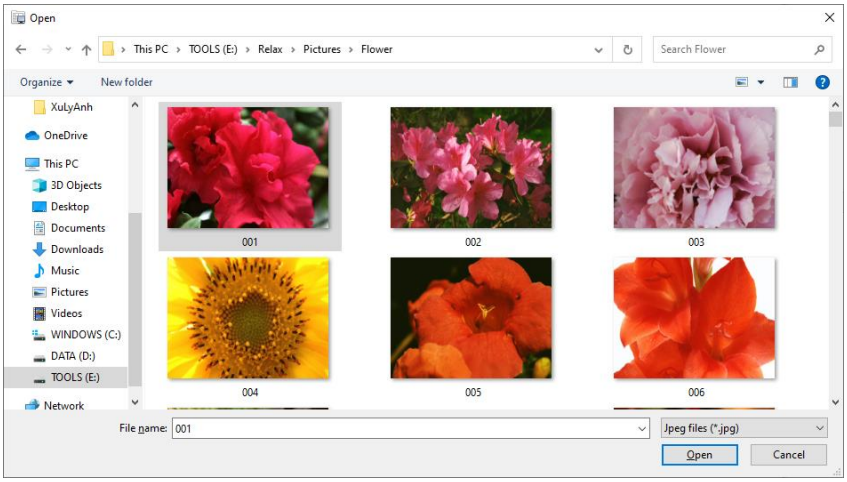


- **Exit:**

Thoát chương trình

- **Open:**

Mở hình ảnh để xử lý, khi kích vào chương trình sẽ mở ra cửa sổ chọn ảnh như hình vẽ dưới, chọn hình ảnh cần xử lý sau đó bấm Open



- **Zoom Out:**

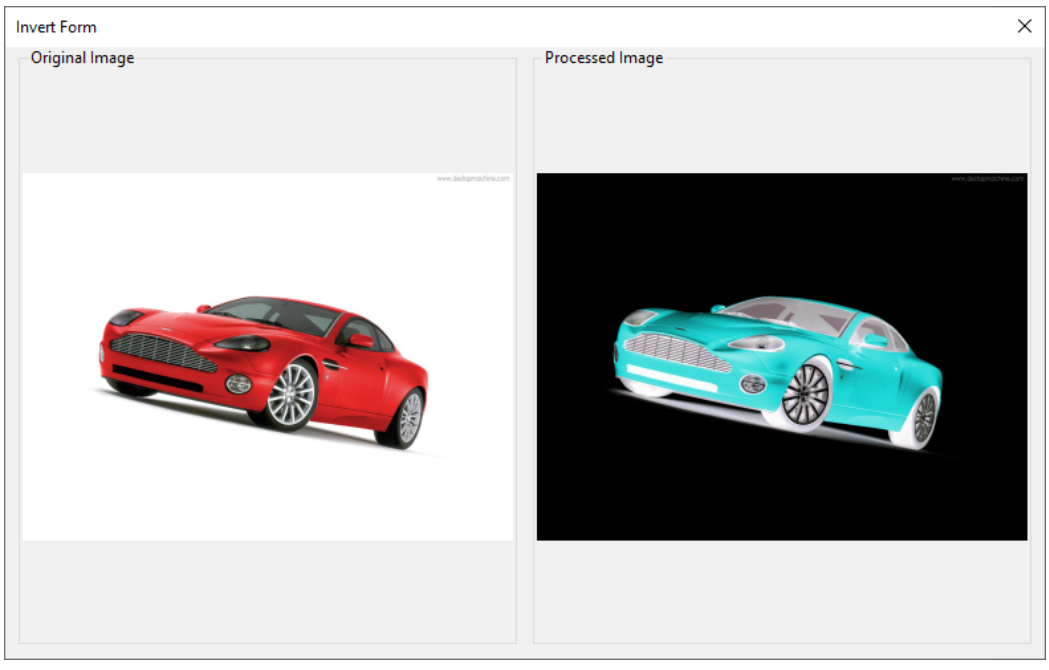
Nếu cần thể hiện chi tiết, bấm vào nút Zoom Out để phóng to hình ảnh, mỗi lần bấm ảnh sẽ được phóng lên với mức 10% so với hiện tại.

- **Zoom In: Thu nhỏ hình ảnh**

Nếu cần thu nhỏ hình ảnh trong chương trình, bấm vào nút Zoom In để thu to hình ảnh, mỗi lần bấm ảnh sẽ được thu nhỏ lại 10% so với hiện tại.

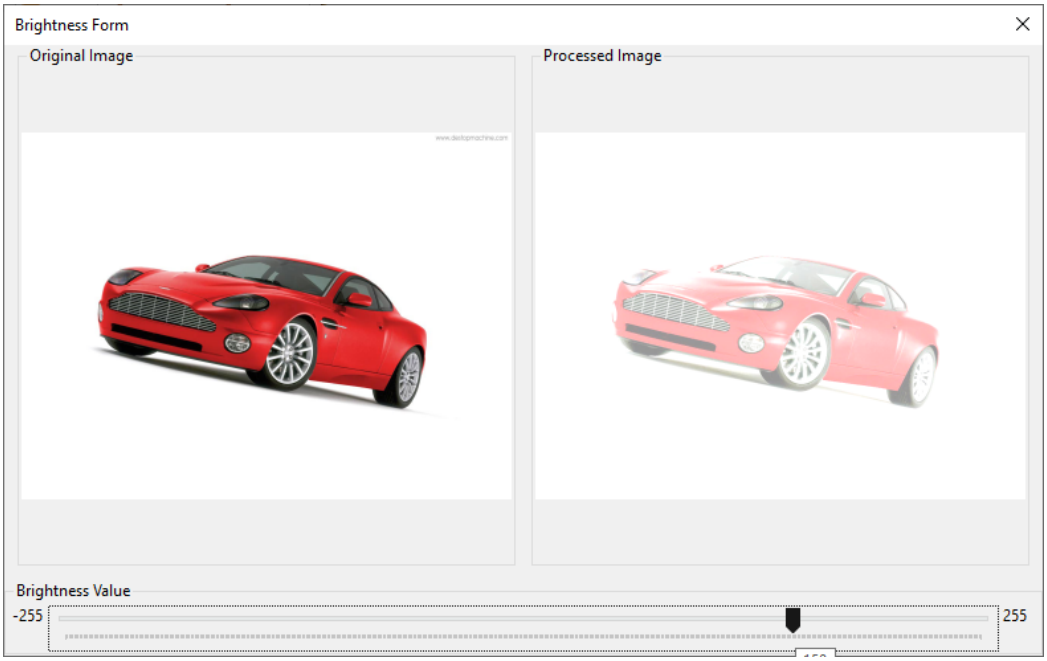
- **Invert: Tạo màu âm bản cho hình ảnh**

Sau khi chọn hình ảnh ở giao diện chính, chọn Filter - > Invert sẽ ra giao diện và kết quả như hình bên dưới.



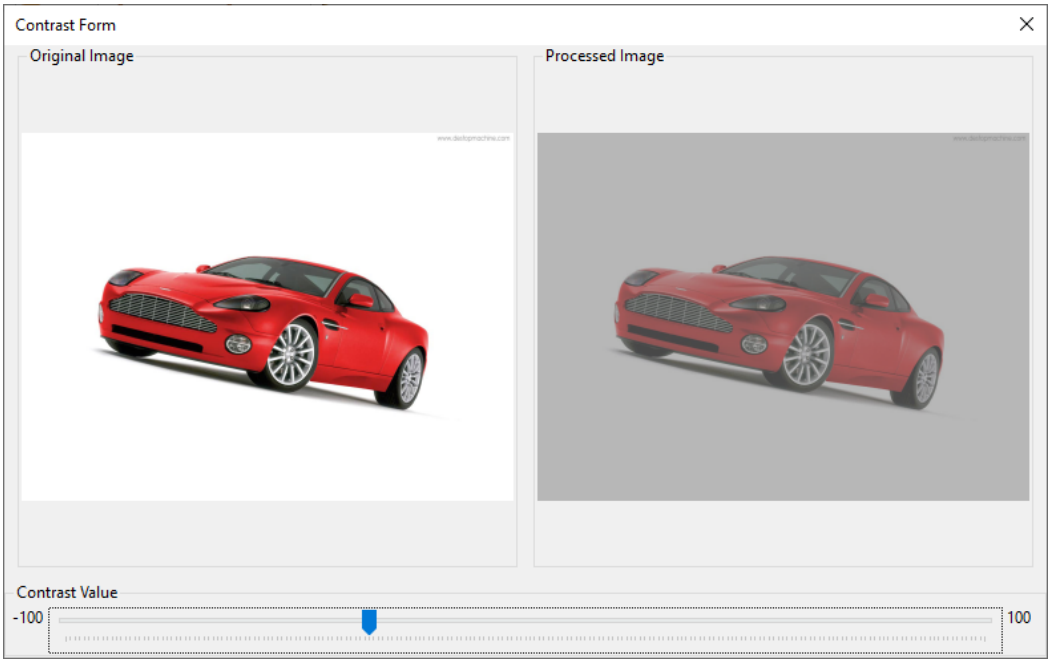
- **Brightness: Hiệu chỉnh ánh sáng hình ảnh**

Sau khi chọn hình ảnh ở giao diện chính, chọn Filter - > Brightness sẽ ra giao diện và kết quả với tham số Brightness=0 như hình bên dưới. Kéo thanh trượt để thay đổi độ sáng của ảnh.



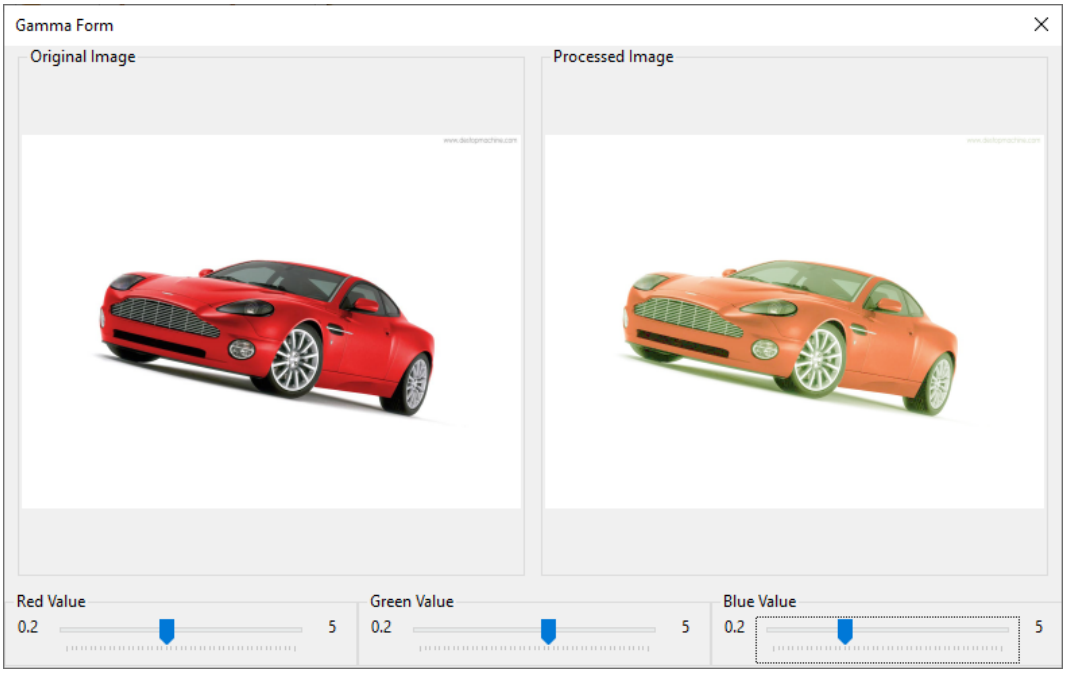
- **Contrast: Thay đổi độ tương phản**

Sau khi chọn hình ảnh ở giao diện chính, chọn Filter - > Contrast sẽ ra giao diện và kết quả với tham số Contrast =0 như hình bên dưới. Kéo thanh trượt để thay đổi độ tương phản của ảnh.



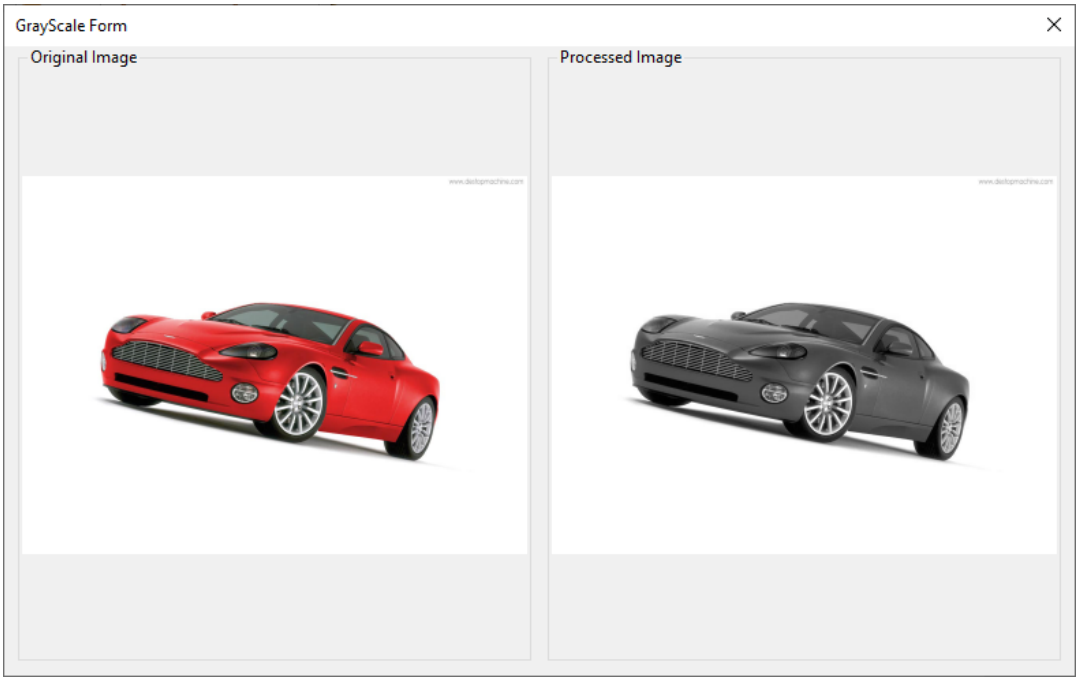
- **Hiệu chỉnh Gamma**

Sau khi chọn hình ảnh ở giao diện chính, chọn Filter - > Gamma sẽ ra giao diện và kết quả với tham số Red=Green=Blue=0.2. Kéo thanh trượt để thay đổi gamma của ảnh.



- **Chuyển ảnh sang đa mức xám (GrayScale)**

Sau khi chọn hình ảnh ở giao diện chính, chọn Filter - > GrayScale sẽ ra giao diện và kết quả như hình vẽ dưới.



TÀI LIỆU THAM KHẢO

[1]. Giáo trình xử lý ảnh. PGS.TS Đỗ Năng Toàn, TS. Phạm Việt Bình

[2]. Giáo trình xử lý ảnh. PGS.TS Nguyễn Quang Hoan