|  |  |  |
| --- | --- | --- |
| TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  Khoa Công nghệ Thông tin  ----------------oOo------------------  BÁO CÁO ĐỒ ÁN GIỮA KỲ  MÔN : Thực hành kỹ thuật lập trình  NĂM HỌC : 2016 – 2017  ĐỒ ÁN : Phần mềm chỉnh sửa ảnh đơn giản (BMP Image Retouch)   |  |  | | --- | --- | | **Nhóm HTH:**  1612861 – Huỳnh Hoàng Huy – Trưởng nhóm  1612858 – Huỳnh Minh Huấn – thành viên  1612889 – Trần Đặng Nhật Tân – thành viên | **Giáo viên phụ trách:**  Thầy Trần Minh Khoa | |

**Mục lục**

[I. Giới thiệu 3](#_Toc483381408)

[II. Bảng phân công công việc, đánh giá mức độ 4](#_Toc483381409)

[III. Tóm tắt đồ án 4](#_Toc483381410)

[IV. Nội dung 5](#_Toc483381411)

[V. Phụ lục và tài liệu tham khảo 12](#_Toc483381412)

# Giới thiệu

1. Sơ lược về BMP:

- **BMP**, còn được biết đến với tên tiếng Anh khác là *Windows bitmap*, là một định dạng tập tin hình ảnh khá phổ biến. Các tập tin đồ họa lưu dưới dạng BMP thường có đuôi là **.BMP** hoặc **.DIB**

- Là tập tin hình ảnh thường không được nén bằng bất kỳ thuật toán nào. Khi lưu ảnh, các điểm ảnh được ghi trực tiếp vào tập tin - một điểm ảnh sẽ được mô tả bởi một hay nhiều byte tùy thuộc vào giá trị **n** của ảnh. Do đó, một hình ảnh lưu dưới dạng BMP thường có kích cỡ rất lớn, gấp nhiều lần so với các ảnh được nén

1. Cấu trúc:

struct BitmapFileHeader {

unsigned short bType; (nhận dạng file bitmap)

unsigned int bSize; (kích thước của file)

unsigned short bReserved1; (biến dự trữ)

unsigned short bReserved2; (biến dự trữ)

unsigned int bOffsets; (vị trí bắt đầu)

};

struct BitmapInfoHeader {

unsigned int bSize; (kích thước phần info)

unsigned int bWidth; (chiều rộng của ảnh)

unsigned int bHeight; (chiều dài của ảnh)

unsigned short bPlanes; (số plane màu)

unsigned short bBitcount; (độ sâu của màu)

unsigned int bCompression; (kiểu nén)

unsigned int bSizeImage; (kích thước ảnh)

unsigned int bXPelsPerMeter; (độ phân giải ngang)

unsigned int bYPelsPerMeter; (độ phân giải dọc)

unsigned int bColorUsed; (số màu sử dụng)

unsigned int bColorImportant; (số màu quan trọng)

};

struct RGBtriple {

unsigned char R, G, B; (kiểu dữ liệu RGB)

};

# Bảng phân công công việc, đánh giá mức độ

|  |  |  |
| --- | --- | --- |
| Họ tên | Công việc | Đánh giá |
| Huỳnh Hoàng Huy | Theo dõi, thúc đẩy tiến độ  Viết báo cáo  Cho ý tưởng  Tìm kiếm thư viện | * Tốt * Tốt * Tốt * Tốt |
| Huỳnh Minh Huấn | Cho ý tưởng  Viết chương trình | * Tốt * Tốt |
| Trần Đặng Nhật Tân | Tìm tài liệu  Hỗ trợ Huấn viết chương trình | * Tốt * Tốt |

# Tóm tắt đồ án

* Đồ án nhóm chúng tôi thực hiện nhằm mục đích thay thế các chương trình hỗ trợ chỉnh sửa ảnh hiện nay.
* Hiện tại chương trình của chúng tôi hỗ trợ 16 công cụ chỉnh sửa cơ bản cho một bức ảnh BMP.
* Có thể trong quá trình thực hiện vẫn còn một số sai sót, mong mọi người thông cảm.

# Nội dung

Init.h

-Sử dụng các thư viện:

+ fstream

+ iostream

+ Windows.h

+ ctime

+ cstdlib

+ cmath

+ string

+ malloc.h

+ stdio.h

-Cấu trúc của file bitmap:

+ BitmapFileHeader

+ BitmapInfoHeader

+ RGBtriple

-Các hàm đồ họa trên console:

+ void GotoXY(int x, int y)

+ void SetColor(int color)

+ void SetForeGoundColor(int color)

+ void ShowCur(bool CursorVisibility)

+ void resizeConsole(int width, int height)

+ void cls()

-Các hàm thiết kế giao diện sử dụng:

+ void ShowIntro()

+ void ShowMenu()

-Hàm đọc và lưu file bitmap:

+ void ReadFileBMP(FILE \*f,BitmapFileHeader &h, BitmapInfoHeader &info, RGBtriple \*\*&p)

+ void SaveFileBMP(FILE \*g, BitmapFileHeader h, BitmapInfoHeader info, RGBtriple \*\*p)

-Các hàm xuất ảnh:

+Hàm xuất ảnh ra console:

* void ShowImgToMenu(RGBtriple \*\*p, int m, int n)
* void ShowDubImg(RGBtriple \*\*p, RGBtriple \*\*q, int m, int n, int m1, int n1)

+Hàm xuất ảnh xoay góc 900:

* + - * void ShowImgFlip90(RGBtriple \*\*p, RGBtriple \*\*q, int m, int n)

+Xuất ảnh cắt

* void ShowCropImg(RGBtriple \*\*p, RGBtriple \*\*q, int m, int n, int m1, int n1)

+Chiếu ảnh ra console

* void ShowImgToConsole(RGBtriple \*\*p, RGBtriple \*\*q, int m, int n)

- Hàm xử lý ảnh, chỉnh ảnh:

+Ảnh xám:

* + - * void GrayScale(RGBtriple \*\*&p, int m, int n)

+Ảnh đen trắng

* + - * void BlackWhite(RGBtriple \*\*p, int m, int n)

+Lật ảnh chiều dọc:

* + - * void FlipVertical(RGBtriple \*\*p, int m, int n)

+Lật ảnh chiều ngang:

* + - * void FlipHorizontal(RGBtriple \*\*p, int m, int n)

+Làm mờ ảnh:

* + - * void BlurImg(RGBtriple \*\*&p, int m, int n);

+Chuyển màu âm bản

* + - * void InvertColor(RGBtriple \*\*&p, int m, int n)

+Làm tối ảnh:

* + - * void Darkening(RGBtriple \*\*&p, int m, int n)

+Làm sáng ảnh:

* + - * void Lightening(RGBtriple \*\*&p, int m, int n)

+Xoay ảnh 900 sang trái:

* + - * void Flip90toLeft(RGBtriple \*\*&p, int &m, int &n)

+Xoay ảnh 900 sang trái phải:

* + - * void Flip90toRight(RGBtriple \*\*&p, int &m, int &n)

+Cắt ảnh theo tọa độ:

* + - * void CropImage(RGBtriple \*\*&p, int &m, int &n)

+Chỉnh độ tương phản:

* + - * void AdjustImage(RGBtriple \*\*p, int m, int n)

+Tạo viền ảnh:

* + - * void BorderImg(RGBtriple \*\*&p, int &m, int &n)

+Nhân bản ảnh theo chiều dọc:

* + - * void DubImgVer(RGBtriple \*\*&p, int &m, int &n)

+Nhân bản ảnh theo chiều ngang:

* + - * void DubImgHor(RGBtriple \*\*&p, int &m, int &n)

+Chỉnh kích thước ảnh:

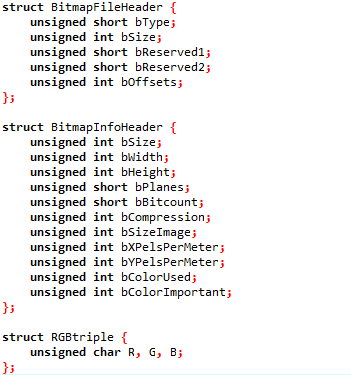
* + - * void ResizeImg(RGBtriple \*\*&p, int &m, int &n)

**Cách thực hiện**

Cấu trúc của một bức ảnh BMP được cố định theo một kiểu xác định:

Các pixel (Red Green Blue)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| BitmapFileHeader | | | | | |
| BitmapInfoHeader | | | | | |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Do đó, để đọc được thông tin của file BMP, mình phải đọc theo thứ tự trên. Chúng ta thiết lặp 3 kiểu dữ liệu mới

Kết hợp với thư viện “init.h”, “iostream ”, “fstream”

Và một số lệnh trong windown: HANDLE wHnd = GetStdHandle(STD\_OUTPUT\_HANDLE), rHnd = GetStdHandle(STD\_INPUT\_HANDLE);

Bây giờ chúng ta cùng lần lượt tìm hiểu về hàm main() mà team đã phát triển

Như đã giới thiệu ở trên: BitmapFileHeader header; BitmapInfoHeader infoheader; RGBtriple \*\*pix = NULL; là 3 thành phần không thể thiếu để đọc file BMP.

Trong hàm ShowIntro(), team đã sử dụng các hàm ẩn hiện con trỏ nhấp nháy (showcur), hàm tạo 1 khoảng thời gian “ngủ” cho chương trình (sleep), nhằm xuất ra màn hình các thông tin giới thiệu của nhóm.

Sau đó, chúng tôi gọi 2 hàm system("pause"); system("cls"); nhằm dừng lại cho người khác đọc, sau đi đọc xong, nhấn phím bất kỳ sẽ xóa màn hình.

Tiếp theo là các thao tác nhập địa chỉ của file BMP. Nếu không hợp lệ phải nhập lại.

Các bước ở trên chỉ là phần “khởi động” thôi! Sau khi đã thiết lặp đẩy đủ công cụ, chúng ta bắt đầu đọc file **ReadFileBMP**(f,header,infoheader,pix). Đọc vào các thông tin cho header và infoheader khá là dễ dàng. Tuy nhiên, khác với 2 cái trên, pixel được lưu trữ là dạng mảng một chiều với những thông số Red Green Blue. Ở đây, để cho dễ thực hiện, chúng tôi sử dụng mảng hai chiều để lưu pixel

Bước tiếp theo, chúng ta sẽ sử dụng 2 biến để lưu chiều dài và chiều rộng của bức hình, và từ đó điều chỉnh màn hình Console.

Tiếp theo, chúng tôi muốn trình bày ra 16 tác vụ có thể thực hiện trong chương trình. Kết hợp với thủ thuật while(1) vòng lập sẽ chạy mãi mãi và sẽ thoát ra ở các lựa chọn switch – case, nếu nhập sai sẽ báo lỗi sau đó nhập lại.

Sau khi thực hiện các lựa chọn, chương trình sẽ hỏi bạn có muốn lưu ảnh lại không (2 biến để lưu chiều dài và chiều rộng sẽ thay đổi để lưu lại hình mới).

Cũng tương tự như đọc vào, **SaveFileBMP** sẽ ghi dữ liệu ra file mới với các thông số mới.

Sau khi thực hiện xong, công việc cuối cùng là hủy con trỏ.

Bây giờ tôi sẽ giới thiệu cho các bạn các chức năng có thể sử dụng được:

1. Lật ảnh theo chiều dọc 180o (FlipVertical)

Chúng ta sẽ tạo ra một con trỏ cấp 2 thuộc kiểu RGBtriple để lưu ngược lại các pixel ở ảnh gốc

1. Lật ảnh theo chiều ngang (FlipHorizontal)

Chúng ta sẽ tạo ra một con trỏ cấp 2 thuộc kiểu RGBtriple để lưu pixel, nhưng sẽ duyệt bắt đầu từ dòng 0 và cột thứ N-1

1. Lật ảnh 90o sang trái (Flip90toLeft)

Ở đây chúng ta sẽ làm ngược lại với phần FlipHorizontal, tức là duyệt từ cột thứ 0 trước và dòng thứ M-1

1. Làm mờ ảnh (BlurImg)

Áp dụng công thức 1 ô sẽ bằng trung bình tổng các ô xung quanh nó và nó

1. Chuyển thành ảnh xám (GrayScale)

Ta cần áp dụng công thức Y = 0.2126 \* R + 0.7152 \* G + 0.0722 \* B

Sau đó gán q[i][j].R = q[i][j].G = q[i][j].B=Y

1. Chuyển thành trắng đen (BlackWhite)

Cũng như GrayScale, nhưng ở đây ta thêm 1 bước, nếu Y>=127 ( tức là 255 div 2 ), thì ta xem nó là màu trắng và gán q[i][j].R = q[i][j].G = q[i][j].B = 255. Ngược lại q[i][j].R = q[i][j].G = q[i][j].B = 0;

1. Tăng độ sáng cho ảnh (Lightening)

Nhập % tăng độ sáng a%

Áp dụng công thức

q[i][j].R = (p[i][j].R\*(1 - a) / 255 + a) \* 255;

q[i][j].G = (p[i][j].G\*(1 - a) / 255 + a) \* 255;

q[i][j].B = (p[i][j].B\*(1 - a) / 255 + a) \* 255;

1. Giảm độ sáng cho ảnh (Darkening)

Giống như Lightening nhưng khác một chút về công thức

q[i][j].R = p[i][j].R\*(1 - a) ;

q[i][j].G = p[i][j].G\*(1 - a);

q[i][j].B = p[i][j].B\*(1 - a);

1. Tăng giảm độ tương phản (AdjustImage)

Nhập tỉ lệ tương phản contrast

Áp dụng công thức: factor = (float)259 \* (contrast + 255) / (255 \* (259 - contrast))

q[i][j].R = truncate((float)(factor \* ((float)p[i][j].R - 128) + 128));

q[i][j].G = truncate((float)(factor \* ((float)p[i][j].G - 128) + 128));

q[i][j].B = truncate((float)(factor \* ((float)p[i][j].B - 128) + 128));

với truncate là hàm làm tròn theo RGB, sẽ trả về value nếu 0<=value<=255 hoặc trả về 0 nếu value<0, trả về 255 nếu value > 255

1. Chuyển màu âm bản (InvertColor)

Áp dụng công thức

q[i][j].R = 255 - p[i][j].R;

q[i][j].G = 255 - p[i][j].G;

q[i][j].B = 255 - p[i][j].B;

1. Cắt ảnh theo tọa độ (CropImage)

B1 : chọn tọa độ (x1, y1) cho ô trái trên và (x2,y2) cho ô phải dưới. Lưu ý : (0 <= x1 < x2 <= m, 0 <= y1 < y2 <= n).

B2 : cho 2 dòng for duyệt từ (x1,y1) -> (x2,y2) để lưu vào mảng động q mới như trong source.

B3 : cập nhật lại : infoheader.bHeight, infoheader.bWidth, header.bSize = infoheader.bHeight \* infoheader.bWidth\*3 + 54 --> ghi dữ liệu ra file out.bmp

1. Tạo viền ảnh (BorderImg)

B1 : chọn màu cho viền (R G B) : ?.

B2 : đổi màu viền sang màu theo (R G B) đã nhập ở B1 (viền là các ô có tọa độ (i,n-1) ,(m-1,j), (0,j), (i,0)).

Lưu ý : vì muốn có viền dày hơn nên tôi tạo viền dày gấp 3 lần theo cách trên.

B3 : các ô ở giữa được sao chép lại như mảng pix gốc truyền vào.

B4 : Lưu lại (hoặc không) và ghi ảnh ra file out.bmp

1. Nhân bản ảnh theo chiều ngang (DubImgHor)

Áp dụng q[i][j] = q[i][j + n] = p[i][j];

1. Nhân bản ảnh theo chiều dọc (DubImgVer)

Áp dụng q[i][j] = q[i + m][j] = p[i][j];

1. Thay đổi kích thước ảnh (ResizeImg)

Nhập tỉ lệ muốn thay đổi (xN)

Sau đó áp dụng:

u = i\*k; v = j\*k;

for (int k1 = 0; k1 < k; k1++)

q[u][v + k1] = q[u + k1][v] = p[i][j];

# Phụ lục và tài liệu tham khảo

1. [Bitmap và các xử lí đơn giản](https://www.stdio.vn/articles/read/523/bitmap-khai-niem-va-cac-xu-ly-co-ban)
2. [Định dạng ảnh Bitmap](https://www.stdio.vn/articles/read/543/dinh-dang-anh-bitmap-gioi-thieu-va-cac-thao-tac-co-ban)
3. [Bitmap là gì?](https://vi.wikipedia.org/wiki/BMP)
4. …

Hết