Lời cảm ơn

Trước hết em xin bày tỏ lòng biết ơn sâu sắc nhất tới thầy giáo Thạc sỹ Lê Thụy đã tận tình giúp đỡ em rất nhiều trong suốt quá trình tìm hiểu nghiên cứu và hoàn thành báo cáo thực tập.

Em xin chân thành cảm ơn sự giúp đỡ rất tận tình của các anh, các chị trong Công ty cổ phần Hà Duy trong quá trình em thực tập tại công ty.

Em xin chân thành cảm ơn các thầy cô trong bộ môn tin cũng như các thầy cô trong trường đã trang bị cho em những kiến thức cơ bản cần thiết để em có thể hoàn thành báo cáo.

Cuối cùng, em xin cảm ơn tất cả các bạn đã động viên, góp ý và trao đổi hỗ trợ cho em trong suốt thời gian vừa qua.

Trong quá trình nghiên cứu và tìm hiểu đề tài. Em sẽ không tránh khỏi nhưng thiếu sót. Em rất mong nhận được sự thông cảm, góp ý và tận tình chỉ bảo của Thầy cô và các

Em xin chân thành cảm ơn!

Hải phòng, ngày 20 tháng 3 năm 2009

Sinh viên

Đặng Đức Hiệp

MỤC LỤC

Lời nói đầu

Ngày nay, cùng với sự phát triển mạnh mẽ của ngành khoa học công nghệ thông tin, internet đã trở thành một nhu cầu, phương tiện không thể thiếu đối với mọi người, nhu cầu trao đổi thông tin qua mạng ngày càng lớn. Và với lượng thông tin lớn như vậy được truyền qua mạng thì nguy cơ dữ liệu bị truy cập trái phép cũng tăng lên vì vậy vấn đề bảo đảm an toàn và bảo mật thông tin cho dữ liệu truyền trên mạng là rất cần thiết. Nhiều kỹ thuật đã được nghiên cứu nhằm giải quyết vấn đề này. Một trong những kỹ thuật quan trọng nhất là mã hóa thông tin. Tuy nhiên một thông điệp bị mã hóa dễ gây ra sự chú ý và một khi các thông tin mã hóa bị phát hiện thì các tin tặc sẽ tìm mọi cách để giải mã.

Một công nghệ mới phần nào giải quyết được những khó khăn trên là giấu thông tin trong các nguồn đa phương tiện như các nguồn âm thanh, hinh ảnh … Xét theo khía cạnh tổng quát thì giấu thông tin cũng là một dạng mật mã nhằm đảm bảo tính an toàn của thông tin, nhưng phương pháp này ưu điểm ở chổ là giảm được khả năng phát hiện ra sự tồn tại của thông tin trong các nguồn mang.

Giấu thông tin là một kỹ thuật còn tương đối mới và đanh phát triển rất nhanh, thu hút được cả sự quan tâm của giới khoa học và giới công nghiệp và cũng còn nhiều thách thức. Nội dung của báo cáo này chủ yếu nghiên cứu về kỹ thuật giấu tin nói chung và giấu tin trong văn bản nói riêng

Chương 1. TỔNG QUAN VỀ KỸ THUẬT GIẤU TIN

1.1 Định nghĩa giấu tin và mục đích của việc giấu tin.

- Giấu tin là kỹ thuật nhúng một lượng thông tin số nào đó vào trong một đối tượng dữ liệu số khác.

Trong quá trình giấu tin để tăng bảo mật, có thể phải dùng *khóa viết mật.* Đó là loại *giấu tin có xử lý*. Nếu không dùng khóa viết mật để Giấu tin, tức là chỉ dấu tin đơn thuần vào môi trường phủ. Đó là loại *Giấu tin đơn thuần*.

- Mục đích của việc giấu tin là đảm bảo an toàn và bảo mật thông tin. Có 2 khía cạnh cần được quan tâm đó là:

+ Bảo mật cho dữ liệu được đem giấu .

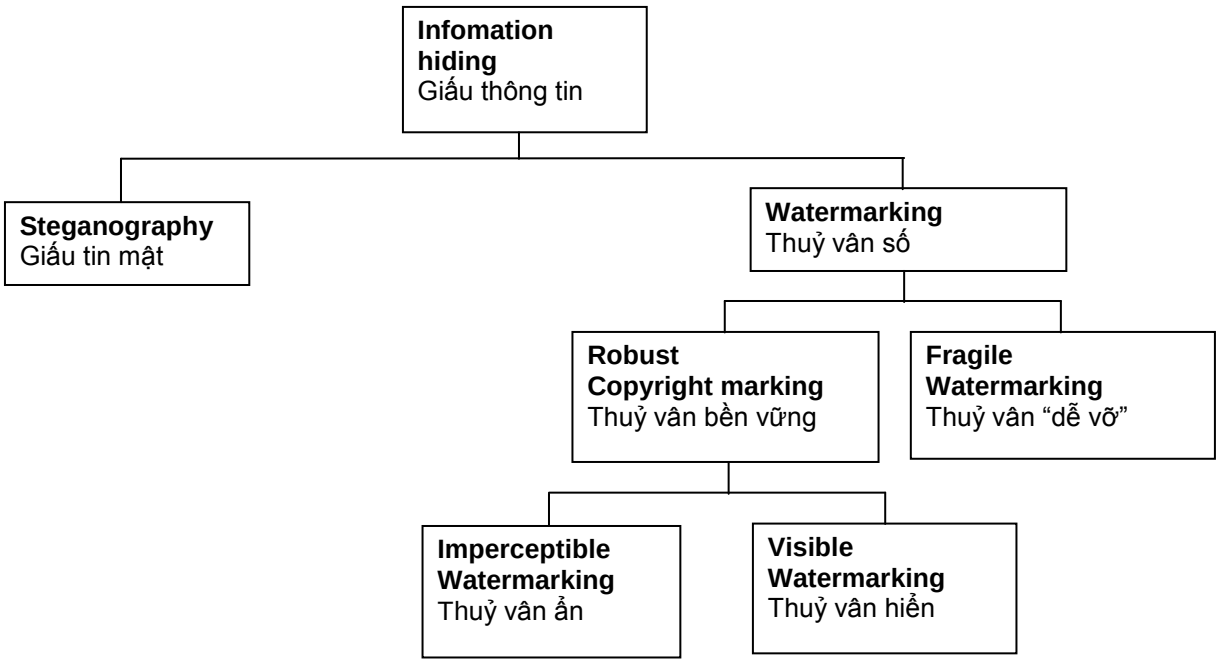
+ Bảo mật cho chính đối tượng được đem giấu thông tin .

- Ngày nay nghệ thuật giấu tin được nghiên cứu để phục vụ các mục đích tích cực như: bảo vệ bản quyền các tài liệu số hóa (dùng thuỷ ấn số), hay giấu các thông tin bí mật về quân sự và kinh tế.

-Sự phát triển của công nghệ thông tin đã tạo ra những môi trường giấu tin mới vô cùng tiện lợi và phong phú. Người ta có thể giấu tin trong các văn bản, hình ảnh, âm thanh. Cũng có thể giấu tin ngay trong các khoảng trống hay các phân vùng ẩn của môi trường lưu trữ như đĩa cứng, đĩa mềm. Các gói tin truyền đi trên mạng cũng là môi trường giấu tin thuận lợi. Các tiện ích phần mềm cũng là môi trường lý tưởng để gài các thông tin quan trọng, để xác nhận bản quyền.

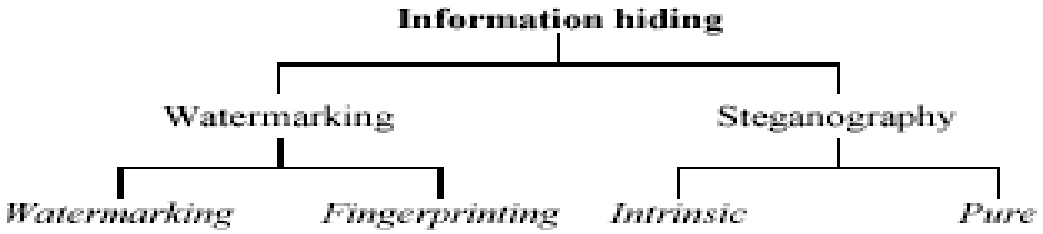
1.2 Phân loại các kỹ thuật giấu tin.

Có thể chia kỹ thuật giấu tin ra làm 2 : steganography và watermarking.



Các lĩnh vực nghiên cứu của mật mã

Trong lĩnh vực bảo mật thông tin, giấu tin bao gồm các vấn đề sau:



Các nhánh của giấu tin

1.2.1 Giấu tin mật (*Seganography*) quan tâm tới việc giấu các tin sao cho thông tin giấu được càng nhiều càng tốt và quan trọng là người khác khó phát hiện được một đối tượng có bị giấu tin bên trong hay không bằng kỹ thuật thông thường.

1.2.2 Thủy vân số (*Watermaking*) đánh giấu vào đối tượng nhằm khẳng định bản quyền sở hữu hay phát hiện xuyên tạc thông tin. Thủy vân số được phân thành 2 loại thủy vân bền vững và thủy vân dễ vỡ.

- *Thuỷ vân bền vững*: thường được ứng dụng trong các ứng dụng bảo vệ bản quyền. Thuỷ vân được nhúng trong sản phẩm như một hình thức dán tem bản quyền. Trong trường hợp này, thuỷ vân phải tồn tại bền vững cùng với sản phẩm nhằm chống việc tẩy xoá, làm giả hay biến đổi phá huỷ thuỷ vân.

+ *Thuỷ vân ẩn*: cũng giống như giấu tin, bằng mắt thường không thể nhìn thấy thuỷ vân.

+ *Thuỷ vân hiện*: là loại thuỷ vân được hiện ngay trên sản phẩm và người dùng có thể nhìn thấy được.

- *Thủy vân dễ vỡ*: là kỹ thuật nhúng thuỷ vân vào trong ảnh sao cho khi phân bố sản phẩm trong môi trường mở nếu có bất cứ một phép biến đổi nào làm thay đổi đối tượng sản phẩm gốc thì thuỷ vân đã được giấu trong đối tượng sẽ không còn nguyên vẹn như trước khi dấu nữa (dễ vỡ).

So sánh giữa steganography và watermarking

	Steganography	**Watermaking**
Mục đích	- Che giấu sự hiện hữu của thông điệp - Thông tin che giấu độc lập với vỏ bọc	-Thêm vào thông tin bản quyền -Che giấu thông tin gắn với đối tượng vỏ bọc
Yêu cầu	Không phát hiện được thông điệp bị che giấu Dung lượng tin được dấu	Tiêu chuẩn bền vững
Tấn công thành công	Phát hiện ra thông điệp bí mật bị che giấu	Watermaking bị phá vỡ

1.3 Một số ứng dụng.

*Ứng dụng của thủy vân số(**Watermaking**):

- Tự động giám sát các bản sao và theo dõi các bản sao, viết tài liệu trên web. (Ví dụ 1 robot tìm kiếm trên web với 1 tài liệu được đánh dấu và do đó có tiềm năng xác đinh vấn đề bất hợp pháp).
- Tự động kiểm tra 1 đài phát thanh truyền đi()

Chương 2. Các đinh dạng ảnh

2.1 Định dạng ảnh BITMAP

Đối tượng ảnh đầu tiên mà các chương trình giấu tin nhắm tới là ảnh Bitmap. Vì ảnh này phổ biến trên mạng Internet, dung lượng giấu tin cao và các phương pháp giấu tin đơn giản.

2.1.1 Tổng quan

Các ảnh số thường được lưu dưới dạng tệp ảnh 24-bit hay 8-bit cho một điểm ảnh. Ảnh 24-bit còn được gọi là ảnh *true colour* cung cấp nhiều chỗ giấu thông tin hơn; tuy nhiên ảnh 24-bit lớn, ví dụ một ảnh 24-bit cỡ 1024 x 768 pixels có kích thước trên 2 MB, nên dễ bị gây chú ý khi tải qua mạng. Thường những ảnh đó cần được nén, nhưng nén ảnh có thể làm mất tin mật.

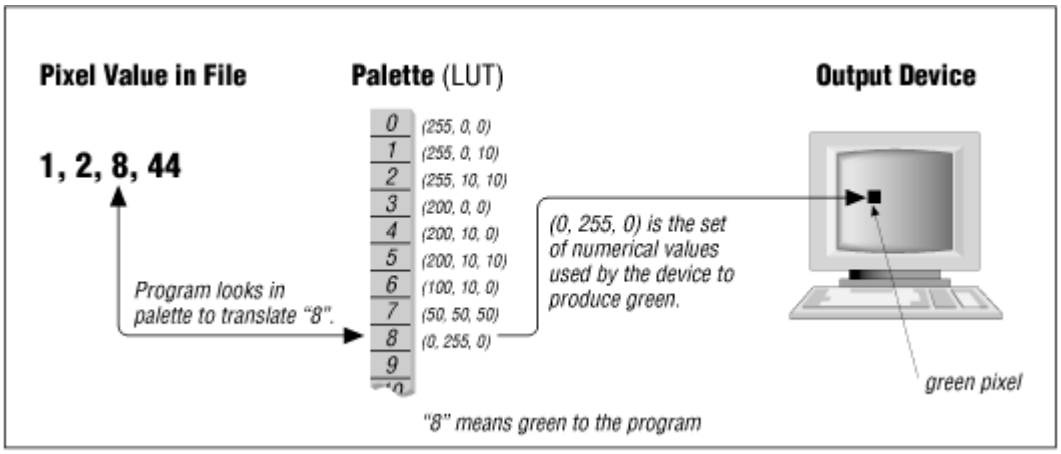
Một phương án khác là có thể dùng ảnh 8-bit mầu để giấu thông tin. Trong các ảnh 8-bit (như ảnh GIF), mỗi điểm ảnh được thể hiện bằng một byte. Mỗi điểm đơn thuần trỏ đến một bảng chỉ mục các mầu (*palette)*, với 256 khả năng mầu. Điểm ảnh chứa trị nằm giữa 0 và 255. Các phần mềm chỉ đơn thuần vẽ mầu cần biểu thị lên màn hình tại vị trí lựa chọn.

Nếu dùng một ảnh 8-bit làm ảnh phủ, rất nhiều chuyên gia về giấu tin trong ảnh khuyên nên dùng ảnh 256 cấp xám vì bảng mầu của ảnh xám thay đổi đồng đều giữa làm tăng khả năng giấu tin.

Giấu tin trong ảnh 8-bit cần xem xét cả ảnh lẫn bảng mầu. Một ảnh có khối lớn các mầu đồng nhất kho giấu hơn vì dễ bị nhận biết. Sau khi chọn ảnh phủ, bước tiếp theo là chọn phương pháp mã hoá ảnh.

2.1.2 Bảng mầu

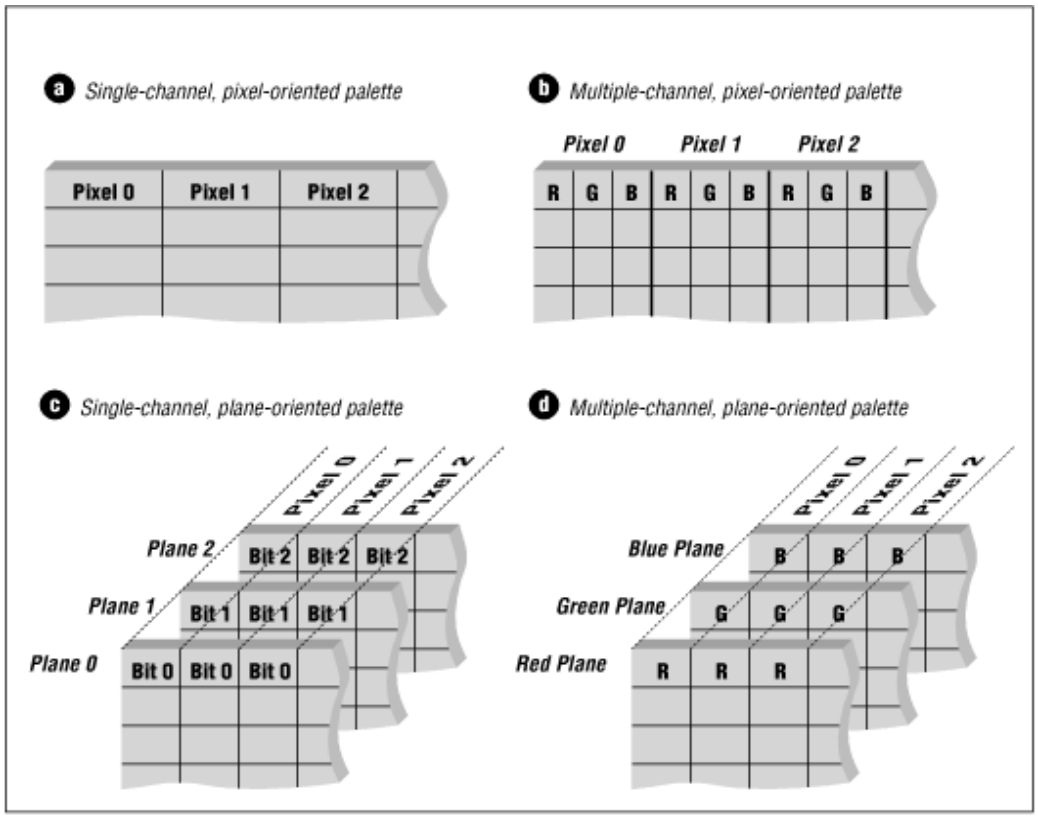
Bảng mầu là một mảng 1 chiều chứa chỉ mục các mầu của ảnh. Sau đó mỗi điểm ảnh chỉ việc trỏ đến một mầu chỉ mục nào đó trên bảng mầu.



Hình 1: Bảng mầu và các điểm ảnh dùng bảng mầu

Trong bảng mầu, 1 mầu ứng với một bộ ba hay bộ bốn
Kích thước của bảng mầu được tính từ độ sâu điểm ảnh (*pixel depth*):
4-bit pixel: 3 byte/mầu * 16 (= 2^4) mầu = 48 byte
8-bit pixel: 3 byte/mầu * 256 mầu = 768 byte
15-bit pixel: 3 byte/mầu * 32768 mầu = 96 kbyte
16-bit pixel: 3 byte/mầu * 65536 mầu = 192 kbyte
Một số loại ảnh giảm bớt số mầu trong bảng mầu, vì không phải tất cả các mầu được dùng trong ảnh (CGM, TGA).
Các giá trị điểm được cất trong 2 byte (16 bit):
16 bit = 2 byte = (8 bit, 8 bit) -> (5,6,5) = (R,G,B)
Có các cách bố trí:

- **Theo điểm ảnh (pixel-orientiert)** (RGB) (RGB) (RGB) (RGB) (RGB)
- **Theo mặt phẳng mầu (plane-orientiert)** (RRRRR ..... GGGGG ..... BB BB B)



Hình 2: Các cách bố trí bảng mầu

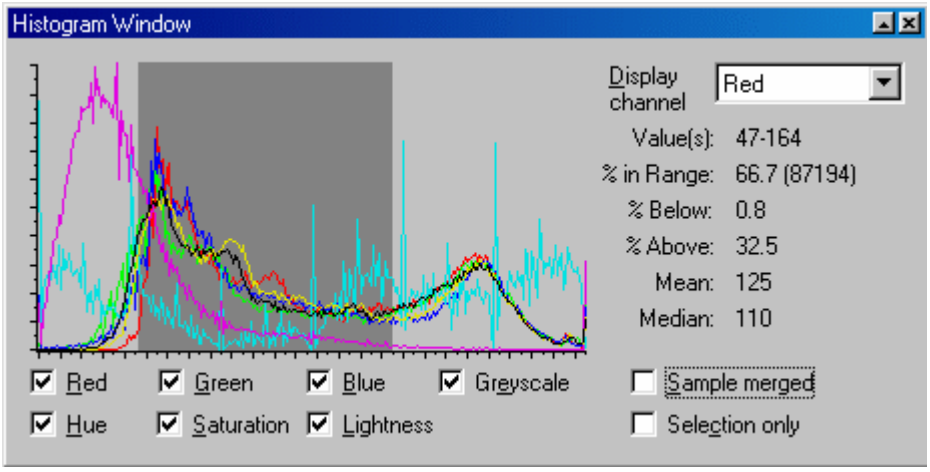
Một số phương pháp giấu tin trong ảnh dựa vào việc sắp xếp lại bảng mầu, trong khi các phương pháp khác thêm bớt các mầu vào bảng mầu.

2.1.3 Mô tả ảnh

Để xử lý hoặc nghiên cứu về ảnh người ta phải mô hình hoá chúng. Tuỳ theo quan điểm, mô hình mà có thể áp dụng các phép xử lý khác nhau trên mô hình đó.

- **Ảnh như một bản đồ bít: q**uan điểm ảnh mành như một bản đồ các bít tạo nền tảng để chúng ta áp dụng các phép toán về bit.
- **Ảnh như một hàm toán học**: để xử lý ảnh trong máy tính dùng các công cụ toán học, người ta tìm cách biểu diễn ảnh như là một hàm rời rạc f(x,y) trong đó x, y là toạ độ của điểm ảnh còn f là giá trị xám hoặc độ sáng của ảnh. f nhận các giá trị rời rạc trong khoảng từ 0 đến f_{max}. Trong ảnh 8 bit thì $f_{max} = 2^8 = 256$. Trong ảnh mầu người ta có thể mô tả mầu qua ba hàm biểu diễn các thành phần đỏ, lục và lam. Ví dụ r(x,y); g(x,y); b(x,y).
- **Ảnh như một môi trường vật lý**: Một ảnh f(x,y) cũng là một môi trường vật lý nên có thể dùng áp dụng các phép biến đổi vật lý trên ảnh. Ví dụ mức năng lượng của điểm ảnh, dải tần số của nhiễu ảnh, dải phổ,..
- **Mô tả ảnh như một mô hình thống kê**: Các giá trị của điểm ảnh (mức xám, độ sáng hay trị mầu) được coi như là biến ngẫu nghiên, do đó chúng ta có thể tính được phân bố xác suất của chúng. Ví dụ người ta có thể dùng biểu đồ cột (histogram) để biểu diễn độ xám hay các trị mầu. Trong xử lý ảnh người ta có thể dùng biểu đồ cột để làm các việc như lọc nhiễu.

Còn trong giấu tin thì ta có thể qua đó mà biết đâu là vùng ảnh có thể giấu tin tốt nhất.



Hình 3: Biểu đồ cột của một ảnh trong Paint Shop Pro 7

Chính các quan điểm khác nhau về ảnh đã làm nền tảng để có được những kỹ thuật khác giấu tin khác nhau

2.1.4 Cấu trúc ảnh

Ảnh Bitmap do Microsoft phát triển, do vậy còn được gọi là **Microsoft Windows Bitmap** (BMP, DIB, Windows BMP, Windows DIB, Compatible Bitmap) được lưu trữ độc lập với thiết bị hiển thị (DIB). Ảnh này được sử dụng rộng rãi trên Windows. Có thể có 1-, 4-, 8-, 16-, 24-, hay 32-bit mầu. Ảnh này thường sử dụng phương pháp mã hoá loạt dài RLE. Kích thước tối đa là 32Kx32K và 2Gx2G pixel. Ảnh bitmap không cho phép chứa nhiều ảnh trong một tệp.
Cấu tạo của ảnh bitmap gồm các phần

1. Header
2. Palette
3. Bitmap Data
4. Footer

Để đọc và xử lý ảnh Bitmap người ta cần nắm được các cấu trúc của ảnh được lưu trong phần Header. Ví dụ Header của Microsoft Windows Bitmap Version 1.x có cấu trúc như sau:

- Header

- Palette
- Bitmap Index
- Palette 1
- File Identifier
- File Version
- Number of Lines per Image
- Number of Pixels per Line
- Number of Bits per Pixel
- Number of Color Planes
- Compression Type
- X Origin of Image
- Y Origin of Image
- Text Description
- Unused Space

Ảnh Bitmap Microsoft Windows 1.x có phần Header gồm 10-byte :

```
TYPEDEF struct _Win1xHeader
{
        WORD Type; /* File type identifier (always 0) */
        WORD Width; /* Width of the bitmap in pixels */
        WORD Height; /* Height of the bitmap in scan lines */
        WORD ByteWidth; /* Width of bitmap in bytes */
        BYTE Planes; /* Number of color planes */
        BYTE BitsPerPixel; /* Number of bits per pixel */
} WIN1XHEADER;
```

Dữ liệu ảnh Bitmap được ghi vào tệp theo 2 cách:

- Quét từng dòng theo trật tự điểm ảnh như chúng được hiển thị trên thiết bị
- Theo từng mặt phẳng mầu (plane)

❖ Chi tiết cấu trúc các ảnh BITMAP

- **Bitmap header:**
  1-2 Nhận dạng file Kiểu arrayp1..2] of char:chứa ký tự BM
  3-6 Kích thướ file Kiểu Longint: tính bằng byte
  7-10 Reserve nt : tôi chưa biết(có lẽ là tên file thừa)
  11-14 Byte bắt đầu Kiểu longint, vị trí byte bắt đầu vùng data kể từ đầu file
- BitmapInfor
  1-4 Số byte trong vùng info Kiểu Longint, hiện tại có giá trị 40
  5-8 Chiều rộng bitmap Kiểu longint tính bằng pixel
  9-12 Chiều cao bitmap Kiểu longint tính bằng pixel
  13-14 Số Planes màu Kiểu Word số bảng màu
  15-16 Số bits cho một pixel Kiểu Word, các giá trị có thể có 1: Đen/trắng, 4:16 màu, 8:256 màu, 24: 24bit màu
  17-20 Kiểu nén dữ liệu Kiểu Longint có giá trị là
  0: Không nén
  1: Nén runlength+8bit/pixel
  2: Nén runlength+4bit/pixel
  21-24 Kích thước ảnh Kiểu Longint, bằng số byte của ảnh
  25-28 Độ phân giải ngang Kiểu Longint, tính bằng pixel
  29-32 Độ phân giải dọc Kiểu Longint, tính bằng pixel
  33-36 Số màu được sử dụng Kiểu Longint trong ảnh
  37-40 Số màu được sử dụng Kiểu Longint khi hiện ảnh
- Bitmap palette
  Tiếp theo sau vùng info là palette màu của BMP, gồm nhiều bộ có kích thước bằng 4 byte xếp liền nhau theo cấu trúc Blue-Green-Red và một Byte dành riêng cho Itensity. Kích thước của vùng Palette màu bằng 4*số màu của ảnh. Vì Palette màu của màn hình có cấu tạo theo thứ tự Red-Green-Blue, nên khi đọc palette màu của ảnh BMP vào ta phải chuyển đổi lại cho phù hợp. Số màu của ảnh được biết dựa trên số bít cho 1 pixel cụ thể là:
  8.bits/pixel: ảnh 256 màu, 4bits/pixel: ảnh 16 màu, 24bits/pixel ảnh 24 bít màu
  BitmapData:
  Phần này kề tiếp ngay sau Palette màu của BMP. Đây là phần chứa các giá trị màu của các điểm ảnh trong BMP. Các điểm ảnh

được lưu theo thứ tự từ trái qua phải trên một dòng và các dòng lại được lưu theo thứ tự dưới lên trên. Mỗi Byte trong vùng BitmapData biểu diễn 1 hoặc nhiều điểm ảnh tùy theo số bits cho một pixel.

- Khi là 1 bit mầu
  Các bitmap là Đơn sắc, và bảng màu có chứa hai mục. Mỗi bit trong bitmap mảng đại diện cho một điểm ảnh. Nếu bit, rõ ràng, các điểm ảnh sẽ được hiển thị với màu sắc của các mục đầu tiên trong bảng màu, nếu các bit, được thiết lập, các điểm ảnh có màu sắc của các mục nhập thứ hai trong bảng.
- Khi là 4 bit mầu.
  Các bitmap đã có tối đa là 16 màu sắc, và các bảng màu chứa lên đến 16 mục. Mỗi điểm ảnh trong bitmap được thể hiện bằng một 4-bit, chỉ mục vào các bảng màu. Ví dụ, nếu là người đầu tiên byte trong bitmap là 1Fh, các byte đại diện cho hai pixel. Đầu tiên chứa các điểm ảnh màu trong bảng màu mục nhập thứ hai, và lần thứ hai chứa các điểm ảnh màu trong bảng màu 16 mục.
- Khi là 8 bit mầu.
  Các bitmap đã có tối đa là 256 màu sắc, và các bảng màu chứa tối đa 256 mục. Trong trường hợp này, mỗi byte trong mảng đại diện cho một điểm ảnh.
- Khi là 16 bit mầu.

Các bitmap đã có tối đa là 2^{16} màu. Nếu các lĩnh vực của nén tập tin bitmap được thiết lập để BI_RGB, các lĩnh vực Palette không chứa bất kỳ mục. Mỗi từ trong mảng bitmap đại diện cho một điểm ảnh. Các thân nhân của intensities đỏ, xanh, xanh và được đại diện với 5 bit cho mỗi thành phần màu sắc. Các giá trị cho màu xanh là đáng kể trong ít nhất 5 bit, sau 5 bit cho mỗi màu xanh và đỏ, tương ứng. Trọng nhất không phải là ít được sử dụng.

Nếu các lĩnh vực của nén tập tin bitmap được thiết lập để BI_BITFIELDS, các lĩnh vực Palette có chứa ba dword màu mặt nạ mà chỉ định màu đỏ, màu xanh, màu xanh và các thành phần, tương ứng, trong mỗi điểm ảnh. Mỗi từ trong mảng bitmap đại diện cho một điểm ảnh.

Windows NT, cụ thể: Khi nén lĩnh vực được thiết lập để BI_BITFIELDS, thiết lập bit trong mỗi dword mask phải được tác và không nên chồng chéo các bit của một khách mask. Tất cả các bit trong các điểm ảnh không cần phải được sử dụng.

Windows 95 cụ thể: Khi nén lĩnh vực được thiết lập để BI_BITFIELDS, Windows 95 chỉ hỗ trợ sau đây 16bpp màu mặt nạ: Một 5-5-5 16-bit, hình ảnh, nơi mà màu xanh mask là 0x001F, các

màu xanh lá cây mask là 0x03E0, và màu đỏ mask là 0x7C00; và 5-6-5 16-bit, hình ảnh, nơi mà màu xanh mask là 0x001F, các màu xanh lá cây mask là 0x07E0, và màu đỏ là 0xF800 mask.

- Khi là 24 bit mầu.

Các bitmap đã có tối đa là 2^{24}màu sắc, và các lĩnh vực Palette không chứa bất kỳ mục. Mỗi 3-byte ban tam ca trong mảng bitmap đại diện cho thân nhân của intensities màu xanh, màu xanh lá cây, và đỏ, tương ứng, cho một điểm ảnh.

- Khi lĩnh vực này là bằng 32.

Các bitmap đã có tối đa là 2^{32}màu. Nếu các lĩnh vực nén của bitmap được thiết lập để BI_RGB, các lĩnh vực Palette không chứa bất kỳ mục. Mỗi dword trong mảng bitmap đại diện cho thân nhân của intensities màu xanh, màu xanh lá cây, và đỏ, tương ứng, cho một điểm ảnh. Cao byte trong mỗi dword là không sử dụng.
Nếu các lĩnh vực nén của bitmap được thiết lập để BI_BITFIELDS, các lĩnh vực Palette có chứa ba dword màu mặt nạ mà chỉ định màu đỏ, màu xanh, màu xanh và các thành phần, tương ứng, trong mỗi điểm ảnh. Mỗi dword trong mảng bitmap đại diện cho một điểm ảnh.

Windows NT, cụ thể: Khi nén lĩnh vực được thiết lập để BI_BITFIELDS, thiết lập bit trong mỗi dword mask phải được tác và không nên chồng chéo các bit của một khách mask. Tất cả các bit trong các điểm ảnh không cần phải được sử dụng.

Windows 95 cụ thể: Khi nén lĩnh vực được thiết lập để BI_BITFIELDS, Windows 95 chỉ hỗ trợ sau đây 32bpp màu mask: Các màu xanh mask là 0x000000FF, các màu xanh lá cây mask là 0x0000FF00, và màu đỏ là mask 0x00FF0000.

2.2 Định dạng ảnh JPEG

JPEG viết tắt của Joint Photographic Experts Group, một nhóm các nhà nghiên cứu đã phát minh ra định dạng này để hiển thị các hình ảnh đầy đủ màu hơn (full-colour) cho định dạng di động mà kích thước file lại nhỏ hơn. Giống như ảnh GIF, JPEG cũng được sử dụng rất nhiều trên Web. Lợi ích chính của chúng hơn GIF là chúng có thể hiển thị các hình ảnh với màu chính xác true-colour (chúng có thể lên đến 16 triệu màu), điều đó cho phép chúng được sử dụng tốt nhất cho các hình ảnh chụp và hình ảnh minh họa có số lượng màu lớn.

Các ảnh JPEG không thể làm trong suốt hoặc chuyển động - trong trường hợp này bạn sẽ sử dụng định dạng GIF (hoặc định dạng PNG để tạo trong suốt).

Tạo ảnh JPEG Fast-Loading: Giống như với các ảnh GIF, để tạo hình JPEG nhỏ đến mức có thể (tính theo bytes) để website tải nhanh hơn. Điều chỉnh chính để thay đổi kích thước file JPEG được gọi là quality, và thường có giá trị từ 0 tới 100%, khi 0% thì chất lượng là thấp nhất (nhưng kích thước file là nhỏ nhất), và 100% thì chất lượng cao nhất (nhưng kích thước file là lớn nhất). 0% chất lượng JPEG sẽ nhìn rất mờ khi so sánh với ảnh gốc. Còn 100% chất lượng JPEG thường không phân biệt được so với ảnh gốc.

Mức độ nhạy cảm của mắt người:

Trong không gian màu YUV, nhãn thị của con người rất nhạy cảm với thành phần Y và kém nhạy cảm với hai loại U và V. Phương pháp nén JPEG đã nắm bắt phát hiện này để tách những thông tin thừa của ảnh. Hệ thống nén thành phần Y của ảnh với mức độ suy giảm ít hơn so với U, V, bởi người ta ít nhận thấy sự thay đổi của U và V so với Y.

Mã hóa:

Công đoạn chính là chia nhỏ bức ảnh thành nhiều vùng nhỏ (thông thường là những vùng 8x8 pixel) rồi sử dụng biến đổi cosin rời rạc để biến đổi những vùng thể hiện này thành dạng ma trận có 64 hệ số thể hiện "thực trạng" các pixel. Điều quan trọng là ở đây hệ số đầu tiên có khả năng thể hiện "thực trạng" cao nhất, khả năng đó giảm rất nhanh với các hệ số khác. Nói cách khác thì lượng thông tin của 64 pixels tập trung chủ yếu ở một số hệ số ma trận theo biến đổi trên. Trong giai đoạn này có sự mất mát thông tin, bởi không có biến đổi ngược chính xác. Nhưng lượng thông tin bị mất này chưa đáng kể so với giai đoạn tiếp theo. Ma trận nhận được sau biến đổi cosin rời rạc được lược bớt sự khác nhau giữa các hệ số. Đây chính là lúc mất nhiều thông tin vì người ta sẽ vứt bỏ những thay đổi nhỏ của các hệ số. Như thế khi bung ảnh đã nén ta sẽ có được những tham số khác của các pixel. Các biến đổi trên áp dụng cho thành phần U và V của ảnh với mức độ cao hơn so với Y (mất nhiều thông tin của U và V hơn). Sau đó thì áp dụng phương pháp mã hóa của Hoffman: Phân tích dãy số, các phần tử lặp lại nhiều được mã hóa bằng ký hiệu ngắn (marker). Khi bung ảnh người ta chỉ việc làm lại các bước trên theo quá trình ngược lại cùng với các biến đổi ngược.

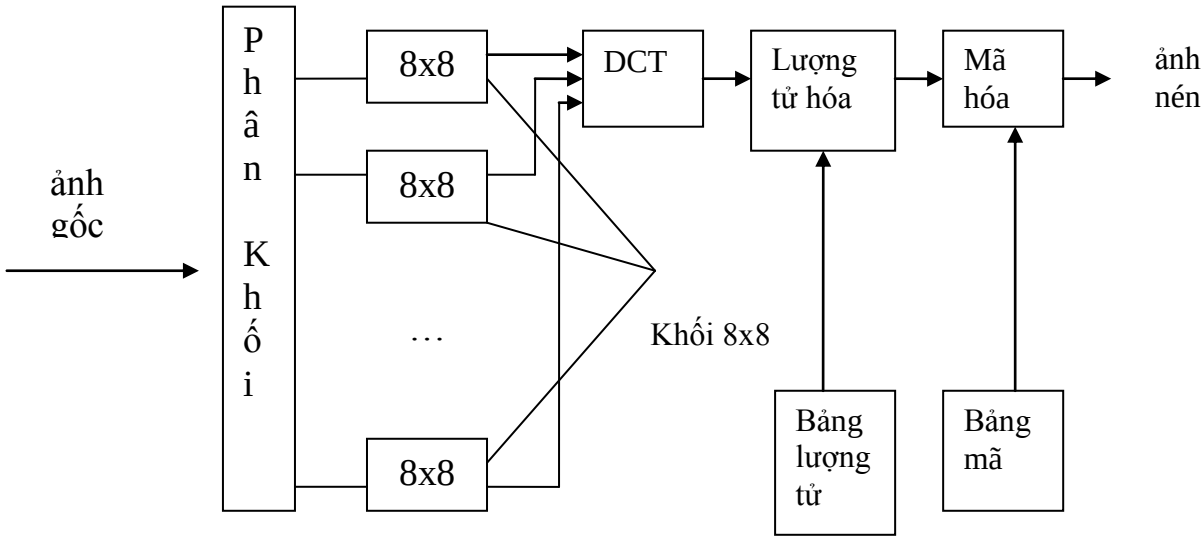
Nhược điểm:

Nhược điểm chính của định dạng JPEG là chúng được nén bằng thuật toán lossy (mất dữ liệu). Điều này có nghĩa rằng hình ảnh của bạn sẽ bị mất một số chi tiết khi chuyển sang định dạng JPEG. Đường bao giữa các khối màu có thể xuất hiện nhiều điểm mờ, và các vùng sẽ mất sự rõ nét. Nói một cách khác, định dạng JPEG thực hiện bảo quản tất cả thông tin màu trong hình ảnh đó, tuy nhiên với các hình ảnh chất lượng màu cao high-colour như hình ảnh chụp thì điều này sẽ không hề hấn gì.

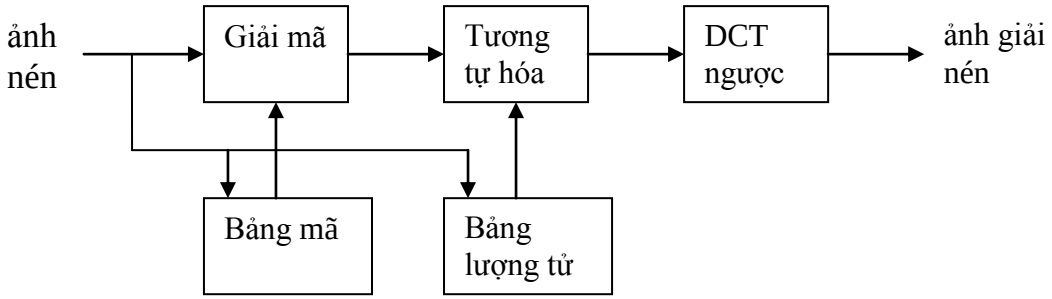
Cấu trúc: JPEG là viết tắt của Joint Photographic Expert Group (nhóm các chuyên gia phát triển chuẩn ảnh này). Chuẩn JPEG được công nhận là chuẩn ảnh quốc tế năm 1990 phục vụ các ứng dụng truyền ảnh cho các lĩnh vực như y học, khoa học kỹ thuật, ảnh nghệ thuật…

Chuẩn JPEG được sử dụng để mã hóa ảnh đa mức xám, ảnh màu. Nó không cho kết quả ổn định lắm với ảnh đen trắng. Chuẩn JPEG cung cấp giải thuật cho cả 2 loại nén là nén không mất mát thong tin và nén mất mát thong tin. Trong phần dưới đây, chúng tôi trình bày chi tiết về một trong các dạng nén biến đổi chấp nhận mất mát thong tin dung biến đổi Cosin của chuẩn JPEG: Biến đổi Cosin tuần tự (Sequential DTC - Based). Biến đổi Cosin tuần tự là kỹ thuật đơn giản nhấtnhưng được dùng phổ biến nhất và nó đáp ứng được hầu hết các đặc tính cần thiết cho phần lớn các ứng dụng.

Mã hóa JPEG bao gồm nhiều công đoạn như đã nêu. Sơ đồ thuật toán nén và giải nén được mô tả như duới đây.



Quá trình giải nén sẽ được làm ngược lại, người ta giải mã từng phần ảnh nén tương ứng với phương pháp nén đã sử dụng trong phần nén nhờ các thông tin lien quan ghi trong phần header của file nén. Kết quả thu được là hệ số đã lượng tử. Các hệ số này được khôi phục về giá trị trước khi lượng tử hóa bằng bộ tương tự hóa. Tiếp đó đem biến đổi Cosin ngược ta được ảnh ban đầu với độ trung thực nhất định.



Bảng mã và bảng lượng tử trong sơ đồ giải nén được dựng lên nhờ những thong tin ghi trong phần cấu trúc đầu tệp (Header) của tệp ảnh nén. Quá trình nén chịu trách nhiệm tạo ra và ghi lại những thong tin này. Phần tiếp theo sẽ phân tích tác dụng của từng khối trong sơ đồ.

Chuẩn nén JPEG phân ảnh ra các khối 8x8. Công doạn biến đổi nhanh Cosin hai chiều cho các khối 8x8 tỏ ra hiệu quả hơn. Biến đổi Cosincho các khối có cùng kích cỡ có thể giảm được một phần các tính toán chung như việc tính hệ số C_j^i . Khi n=8 chúng ta chỉ cần tính hệ số C_j^i cho 3 tầng ($8=2^3$), số các hệ số là 4+2+1=7

Nếu với một ảnh 1024 x 1024, phép bién đổi nhanh Cosin một chiều theo hàng ngang hoặc hàng dọc ta phải qua 10 tầng ($1024=2^{10}$). Thời gian tính các hệ số C_j^i là 512 + 256 + 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 1021. Thời gian tính các hệ số C_j^i với toàn bộ ảnh 1024 x 1024 lớn gấp 150 lần so với thời gian tính toán các hệ số này cho các khối.

Biến đổi Cosin đối với các khối có kích thước nhỏ sẽ làm tăng độ chính xác khi tính toán với số dấu phẩy tĩnh, giảm thiểu sai số do làm tròn sinh ra.

Do các điểm ảnh hàng xóm có độ tương quan cao hơn,do đó phép biến đổi Cosin cho từng khối nhõe tập trung năng lượng hơn vào một số ít các hệ số biến đổi. Việc loại bớt một số hệ số năng lượng thấp trong các khối chỉ tạo ra mất mát thông tin cục bộ giúp nâng cao chất lượng ảnh.

Ảnh sẽ được chia làm B khối với :

$$B=\left(\frac{M'}{k}\right)\times\left(\frac{N'}{l}\right)=M_B \times N_B$$

Các khối được xác định bởi bộ số (m,n) với m = [0..M_B-l] và n = [0..N_B-l], ở đây m chỉ thứ tự của khối theo chiều rộng, n chỉ thứ tự của khối theo chiều dài. Phân khối thực chất là xác định tương quan giữa tọa độ riêng trong khối với tọa độ thực của điểm ảnh trong ảnh ban đầu. Nếu ảnh ban đầu ký hiệu Image[i,j] thì ma trận biểu diễn khối (m,n) là x[u,v] được tính:

x[u,v] = Image[mk + u , nl +v]

Chương 3 Giấu tin trong ảnh

3. 1 Các kĩ thuật giấu tin trong ảnh BITMAP

Việc giấu tin trong ảnh màu thì có rất nhiều thuận lợi so với việc giấu tin trong ảnh đen trắng nó có một ưu điểm như sau:

- Giấu được nhiều thông tin hơn so với ảnh đen trắng cùng kích cỡ
- Độ an toàn cao hơn so với ảnh đen trắng vì có rất ít sự thay đổi so với ảnh gốc ban đầu.

Đối với từng loại ảnh màu ta lại có kỹ thuật khác nhau:

3. 1. 1 Ảnh nhỏ hơn hoặc bằng 8 bit màu:

Không phải tất cả những ảnh nhỏ hơn hoặc bằng 8 bit màu đều có bảng màu được sắp xếp, do vậy việc sắp xếp LSB rất khó khăn. Ta cần sắp xếp lại bảng màu:

- Chọn một màu bất kỳ giả sử màu có dạng: A(x, y, z) ta đưa vào vị trí đầu tiên
- Duyệt tất cả các màu B(m, n, p) còn lại và tính:

  $S(A, B) = sqrt((x-m)^2 + (y-n)^2 + (z-p)^2)$
- Ta sẽ chọn màu B có S(A, B) nhỏ nhất để sắp xếp cạnh màu A sau đó lại tiếp tục bước 2.
- Quy trình kết thúc khi bảng màu đã được sắp xếp.

Lưu ý: Các điểm ảnh có chỉ số màu là 15 phải được đổi thành chỉ số 85

3. 1. 2 Ảnh 16 bit màu

Thực tế chỉ có 15 bit được dùng để biểu diễn cho một điểm ảnh :

- 5 bit dùng để biểu diễn cường độ tương đối màu đỏ
- 5 bit dùng để biểu diễn cường độ tương đối màu xanh lơ
- 5 bit dùng để biểu diễn cường độ tương đối màu xanh lam

Còn 1 bit không dùng đến là bit cao nhất ở byte thứ 2, đó chính là bit LSB của ảnh 16 bit màu. Nếu chỉ lấy một bit này thì lượng thông tin giấu là rất ít do đó cần lầy thêm một số bit nữa.

3. 1. 3 Ảnh 24 bit màu

Mỗi một điểm ảnh được biểu diễn bằng 3 byte

- 1 byte dùng để biểu diễn cường độ tương đối màu đỏ
- 1 byte dùng để biểu diễn cường độ tương đối màu xanh lơ
- 1 byte dùng để biểu diễn cường độ tương đối màu xanh lam

Trong mỗi byte các bít nằm càng về cuối càng ít ảnh hưởng đến phần dữ liệu ảnh . Thông thường để tăng lượng thông tin được giấu người ta thường lấy 4 bít cuối mỗi byte để giấu thông tin.

Bằng thực nghiệm cho thấy nếu thay toàn bộ bít cuối của một byte thì ảnh kết qủa cũng không khác nhiều lắm so với ảnh ban đầu. Điều này là vô cùng có ý nghĩa vì ta có thể giấu được nhiều thông tin trong ảnh .

3. 1. 4 Các phương pháp giấu tin

- Phương pháp Nhúng vào các bít có trọng số thấp (Least Significant Bit)
- Các phương pháp dựa vào kỹ thuật biến đổi ảnh, ví dụ biến đổi từ miền không gian sang miền tần số
- Các phương pháp sử dụng mặt nạ giác quan

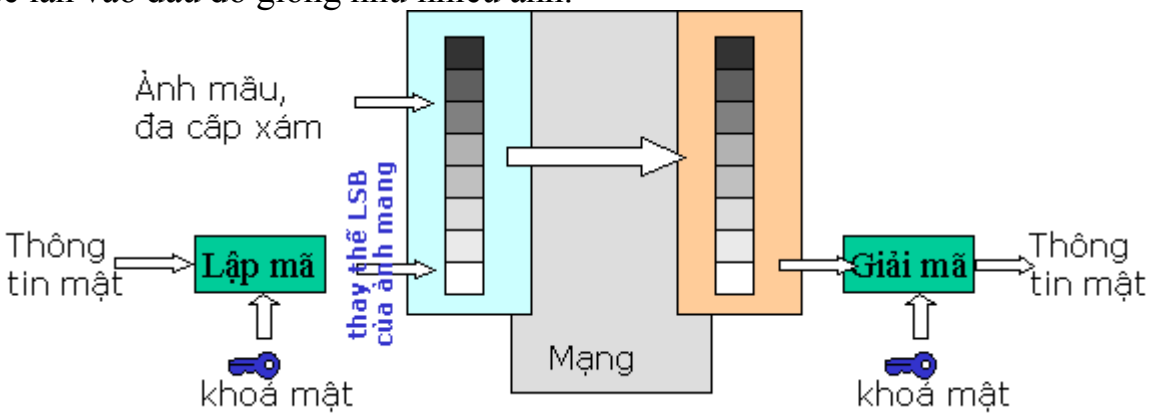
Ví dụ: Kỹ thuật gài vào các bít có trọng số thấp

❖ Nền tảng kĩ thuật:

Khi chuyển một ảnh tương tự sang ảnh số người ta thường chọn ba cách thể hiện mầu:

- 24-bit mầu: mỗi điểm có thể nhận một trong 2^24 mầu, mỗi mầu được tạo từ ba mầu căn bản: red (R), green (G) và blue (B), mỗi mầu nhận một trị từ 0 đến 255 (8 bit)
- 8-bit mầu: mỗi điểm có thể nhận một trong 256 mầu, chọn từ một bảng mầu (palette)
- 8-bit dải xám: mỗi điểm nhận một trong 256 (2^8) sắc thái xám

Phương pháp LSB sửa bít hay các bít có trọng số thấp nhất (ít quan trong nhất để tạo nên mầu điểm ảnh), gài các thông tin mật vào đó. Các thông tin được giấu sẽ lẩn vào đâu đó giống như nhiễu ảnh.



Giấu tin vào các bít ít quan trọng của điểm ảnh

Áp dụng kỹ thuật LSB, một điểm ảnh 24-bit có thể giấu được ba bit thông tin (vì mỗi điểm được thể hiện bằng ba byte). Mọi sự thay đổi trên điểm ảnh có trọng số thấp đều không gây nên sự chú ý của mắt người.

❖ Ví dụ minh họa

Chữ cái A có mã ASCII là 65 hệ thập phân (1000001 hệ nhị phân). Các điểm ảnh trước khi giấu. Để giấu chữ 'A' cần ba điểm ảnh liên tiếp.

Ví dụ các điểm ảnh trước khi giấu là:

00100111 11101001 11001000
00100111 11001000 11101001
11001000 00100111 11101001

Chèn giá trị nhị phân của chữ 'A' vào ba điểm ảnh trên bắt đầu từ byte trên cùng bên trái sẽ cho kết quả:

00100110 11101001 11001000
00100110 11001000 11101000
11001000 00100110 11101001

Các bit được gạch chân là các bit bị lật. Có thể dùng hai bit có trọng số thấp để giấu tin mà chất lượng không thay đổi mấy đối với mắt thường.
Từ ví dụ trên ta có thể suy ra rằng nếu dùng 1 LSB thì xác suất phải lật bít là 50%, vậy nên lượng nhiễu gây ra cho ảnh là rất ít.
Đối với ảnh mầu 24 bit, đôi khi chúng ta có thể dùng đến 2 hoặc thậm chí 3 bít thấp mà vẫn không để lộ thông tin mật. Đối với ảnh 8 bit thì điều này là không thể, và người ta chỉ dùng 1 bít thấp nhất để giấu tin

3. 2 Các kĩ thuật giấu tin trong ảnh JPG

- Kỹ thuật dùng hệ số DCT (JPEG)
- Kỹ thuật mặt nạ và lọc
- Kỹ thuật dùng hệ số của phép chiếu trực giao

3.2.1 Kĩ thuật dùng hệ số DCT :

a) Nền tảng kỹ thuật

Các ảnh JPEG có tỷ lệ nén cao, chất lượng tốt, do đó chúng được sử dụng nhiều trên mạng. Tuy nhiên các tệp ảnh JPEG không phù hợp với xử lý bít như các ảnh dựa trên bảng mầu, tuy vậy vẫn có thể dùng chúng để giấu dữ liệu.

Ảnh JPEG sử dụng biến đổi Cosin rời rạc để thực hiện nén ảnh

Biến đổi cosin rời rạc là phép biến đổi mất dữ liệu vì không thể tính chính xác các giá trị cosin, cũng như có thể có các lỗi làm tròn. Độ lệch giữa dữ liệu gốc và dữ liệu phục hồi lại sau khi biến đổi phụ thuộc vào các giá trị và phương pháp sử dụng để tính các trị cosin rời rạc.

Cũng có thể xử lý ảnh dùng biến đổi Fourier nhanh hoặc biến đổi sóng con (wavelet transformation).

Thuật toán JPEG làm việc bằng cách chia ảnh ra thành các ma trận 8x8. Sau đó tính hệ số biến đổi cosin rời rạc cho từng ma trận. Bước tiếp theo các hệ số này được nhân với một ma trận lượng hoá. Kết quả thu được sẽ được làm tròn đến số nguyên gần nhất, cuối cùng các số nguyên này được nén và lưu lại.

Các cấu tử DCT chính là nơi chúng ta có thể giấu dữ liệu. Cách tiếp cận phổ biến là chọn các hệ số DCT lớn và sử đi chút ít. Vì hệ số lớn tức mức "năng lượng" cao nên ít làm thay đổi ảnh nhất. Một hướng khác là chọn các hệ số DCT trong các vùng mà mắt người không nhìn thấy.

Các thuật toán JPEG nổi tiếng áp dụng trong F5 và JSteg đều dùng cách sửa DCT để nhúng dữ liệu. Cả hai phương pháp này đều qua được mắt thường nhưng không qua được các phương pháp phân tích thống kê.

b) Dung lượng giấu

Dung lượng giấu không cao, và vì vậy phù hợp hơn với thuỷ ấn

c) Phép biến đổi cosin rời rạc:

Biến đổi cosin rời rạc viết tắt là DCT-Discrete Cosine Transform được đưa ra bởi Ahmed và các đồng nghiệp của ông vào năm 1974. Phép biến đổi DCT đã được dùng trong dạng chuẩn ảnh JPEG.

❖ ***Định nghĩa biến đổi cosin rời rạc hai chiều:***

- Biến đổi DCT hai chiều tổng quát là biến đổi trên khối hai chiều bất kỳ MxN, trong đó các khối kích thước 8*8, 16*16 được sử dụng nhiều nhất. Tuy nhiên, chúng ta sẽ chỉ tìm hiểu phép biến đổi DCT trên khối 8*8 được sử dụng trong chuẩn nén ảnh JPEG.

- Phép biến đổi thuận DCT 8*8 được định nghĩa như sau:

$$I(u,v)= \frac{\zeta(u)\zeta(v)}{4}\sum_{k=0}^{7}\sum_{l=0}^{7} X(k,l)\cos(\frac{(2k+1)u\pi}{16})\cos(\frac{(2l+1)v\pi}{16})s$$

I(u.v) được gọi là hệ số DCT và là số thực.

- Biến đổi ngược DCT định nghĩa như sau:

$$X(k,l)= \sum_{k=0}^{7}\sum_{l=0}^{7}\frac{\zeta(u)\zeta(v)}{4} I(u,v)\cos(\frac{(2k+1)u\pi}{16})\cos(\frac{(2l+1)v\pi}{16})$$

ở đây 0<=k,l,u,v<=7 và $\xi(u)= \begin{cases} \frac{1}{\sqrt{2}}(u>0) \\ 1(u=0) \end{cases}$ $\xi(v)= \begin{cases} \frac{1}{\sqrt{2}}(v>0) \\ 1(v=0) \end{cases}$

❖ ***Đặc điểm của phép biến đổi DCT hai chiều:***

- Thể hiện đặc tính nội dung về tần số thông tin ảnh. Hệ số góc trên là lớn và đặc trưng cho giá trị trung bình thành phần một chiều gọi là hệ số DC, còn các hệ số khác có giá trị nhỏ hơn biểu diễn cho các thành phần tần số cao theo hướng ngang và theo hướng thẳng đứng gọi là hệ số AC.

- Bản thân biến đổi DCT không nén được dữ liệu vì cũng sinh ra 64 hệ số.

- Theo nguyên lý chung, khi biến đổi chi tiết giữa các điểm ảnh càng lớn theo một hướng nào đó trong khối các điểm ảnh, hướng ngang, hướng thẳng đứng hay theo hướng đường chéo thì tương ứng theo các hướng đó, các biến đổi DCT càng lớn.

- DCT làm giảm độ tương quan không gian của thông tin trong khối ảnh. Nhờ các đặc tính tần số không gian của hệ thống nhìn của mắt người, các hệ số DCT có thể được mã hoá phù hợp, chỉ các hệ số DCT quan trọng nhất mới được mã hoá để truyền đi.

- Khối hệ số DCT có thể chia thành 3 miền: miền tần số thấp, miền tần số cao, miền tần số giữa:

+ Miền tần số thấp: chứa các thông tin quan trọng ảnh hưởng đến tri giác.

+ Miền tần số cao: các thông tin trong miền tần số cao thường không mang tính tri giác cao, khi nén JPEG thì thường loại bỏ thông tin trong miền này. Trong các thuật toán thuỷ vân, miền hệ số DCT tần số cao thường không được sử dụng do nó thường không bền vững với các phép xử lý ảnh, hoặc nén ảnh JPEG. Miền tần số cao cũng khó được sử dụng do một sự thay đổi dù nhỏ trong miền này cũng dẫn đến chất lượng tri giác của ảnh.

+ Miền tần số giữa: thường hay được sử dụng nhất và cũng cho kết quả tốt nhất. Trong thuật toán đề xuất cũng sử dụng miền tần số ở giữa.

3.2.2 Kỹ thuật giấu tin trong miền biến đổi DCT

Thuật toán dưới đây sẽ sử dụng phương pháp nhúng thuỷ vân trong miền tần số của ảnh, giải tần được sử dụng để chứa tín hiệu thuỷ vân là miền tần số ở giữa của một khối DCT 8x8. Trong đó, các khối DCT 8x8 là những khối ảnh cùng kích thước đã được chọn ra ngẫu nhiên từ ảnh ban đầu và được áp dụng phép biến đổi cosin rời rạc DCT để chuyển sang miền tần số. Mỗi tín hiệu thuỷ vân sẽ được chứa trong một khối.

3.2.2.1 Mô tả thuật toán:

- Input:

  Watermark: Một chuỗi các bit b.

  Một ảnh F.

- Output:

  Một ảnh sau khi thuỷ vân, F'.

  Khoá để giải mã K.

3.2.2.2 Quá trình Watermarking:

Một ảnh có kích m x n sẽ được chia thành (m x n)/64 khối 8 x 8, mỗi bit sẽ được giấu trong một khối.

Chọn một khối bất kỳ B và biến đổi DCT khối đó thu được B'.

Chọn hai hệ số ở vị trí bất kỳ trong miền tần số ở giữa của khối DCT, giả sử đó là b'(i,j) và b'(p,q). Ta tính:

$$d = || b'(i,j) - |b'(p,q)|| \bmod a$$

Trong đó a là một tham số thoả mãn: a= 2(2t + 1), t là một số nguyên dương.

- Bit s_i sẽ được nhúng sao cho thoả mãn điều kiện : $\begin{cases} d >= 2t+1 (s_i = 1) \\ d < 2t+1 (s_i = 0) \end{cases}$

+ Nếu d<2t+1 mà s_i =1 thì một trong hai hệ số DCT b'(i,j) hoặc b'(p,q) có trị tuyệt đối lớn hơn sẽ bị thay đổi để d>= 2t+1 theo công thức sau:

$$\max(|b'(i,j)| , |b'(p,q)|) + (INT(0.75*a)- d)$$

với hàm max (|b'(i,j)| , |b'(p,q)|) là hàm chọn ra hệ số có trị tuyệt đối lớn hơn, hệ số được chọn sẽ được cộng thêm một lượng là (INT(0.75*a) - d) hoặc cũng có thể biến đổi một trong hai hệ số theo công thức:

$$\min(|b'(i,j)| , |b'(p,q)|) - (INT(0.75*a) + d)$$

với hàm min(|b'(i,j)| , |b'(p,q)|) là hàm chọn ra hệ số có trị tuyệt đối nhỏ hơn, hệ số được chọn sẽ bị trừ đi một lượng là (INT(0.75*a) + d)

INT() là hàm làm lấy phần nguyên của một số thực.

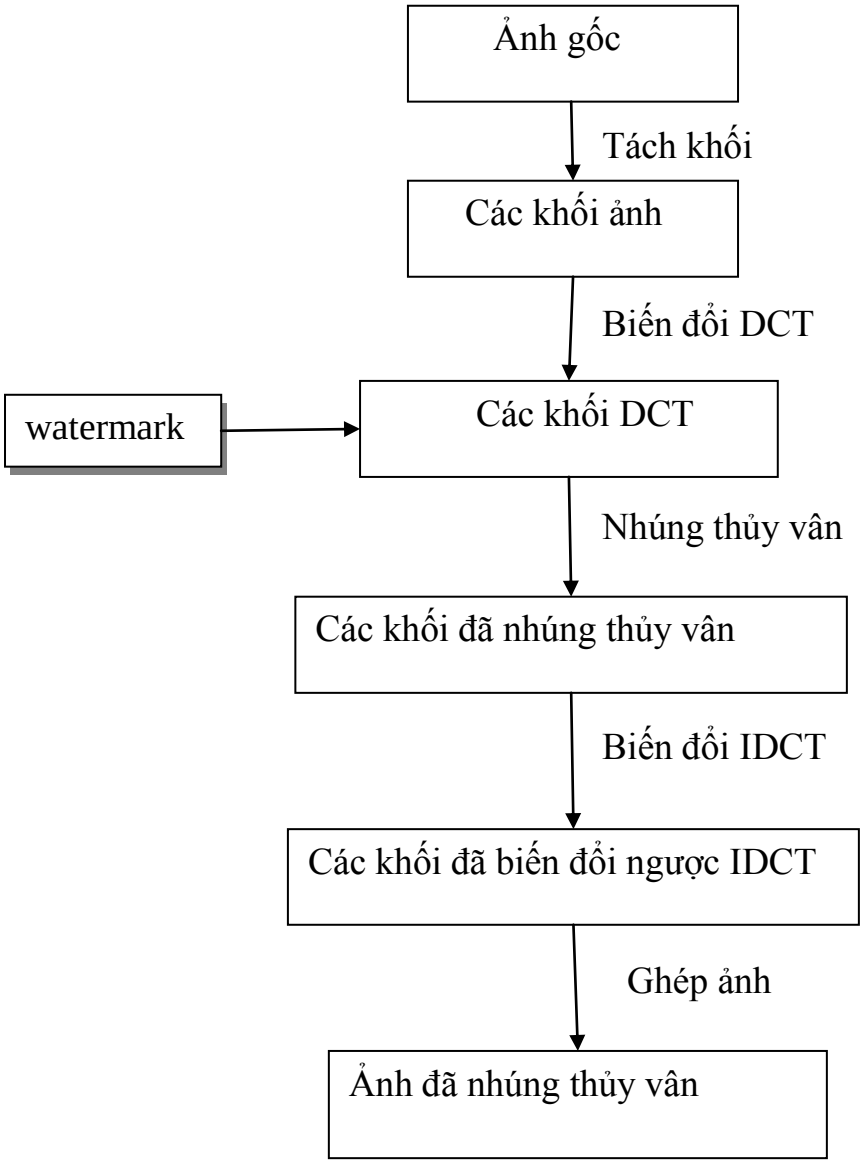
+ Tương tự, nếu d>= 2t+1 mà s_i = 0 thì một trong hai hệ số DCT b'(i,j) hoặc b'(p,q) có trị tuyệt đối lớn hơn sẽ được thay đổi để thoả mãn d<2t+1 như sau:

$$\max(|b'(i,j)| , |b'(p,q)|) – (d- (INT(0.75*a)))$$

hàm max(|b'(i,j)| , |b'(p,q)|) là hàm chọn ra hệ số có trị tuyệt đối lớn, hệ số được chọn sẽ bị trừ đi một lượng là (d- (INT(0.75*a)) hoặc cũng có thể biến đổi một trong hai hệ số theo công thức:

$$\min(|b'(i,j)| , |b'(p,q)|) + (INT(0.75*a)- d)$$

hàm min(|b'(i,j)| , |b'(p,q)|) là hàm chọn ra hệ số có trị tuyệt đối nhỏ hơn, hệ số được chọn sẽ được cộng thêm một lượng là (INT(1.25*a)-d)



- Quá trình nhúng thủy vân được mô tả qua sơ đồ sau:

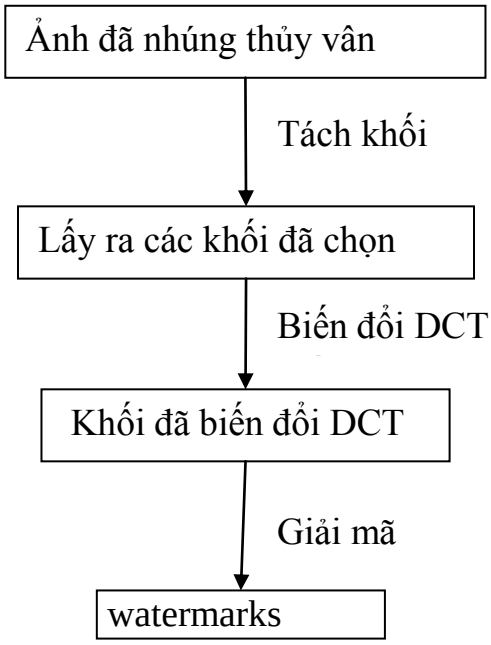
- Quá trình giải nhúng:

Đọc vào khối DCT đã nhúng thủy vân và vị trí hai hệ số đã biến đổi, sau đó tính: d=||b'(i,j)|-|b'(p,q)|| mod a với (a=2(2t+1))

Nếu d>=2t+1 thì s_i =1

Nếu d<2t+1 thì s_i =0

Quá trình giải mã được mô tả như sau:



- Chứng minh tính đúng đắn của thuật toán:

Xét các trường hợp sau đây:

+ Hai trường hợp nếu d< 2t+1 với s_i =0 và d>=2t+1 với s_i =1 thì sẽ không thay đổi gì hệ số của khối DCT, và vì DCT là phép biến đổi hoàn toàn đảo ngược nên khi giải mã thì ta cũng thu được kết quả chính xác.

+ Trường hợp d<2t+1 và s_i =1

Ta biến đổi một trong hai hệ số b'(i,j) và b'(p,q) như sau:

max(|b'(i,j)| , |b'(p,q)|) + (INT(0.75*a)- d)

Khi đó giá trị d mới là :

d' = (||b'(i,j)| -|b'(p,q)| |+ (INT(0.75*a)- d)) mod a

$\Leftrightarrow$ d’ = (|b’(i,j)| - |b’(p,q)|| mod a)+ (INT(0.75*a)mod a)-(d mod a)

$\Leftrightarrow$ d’ = d+INT(0.75*a)-d = INT(0.75*a)> 0.5*a=2t+1(dfcm)

+ Trường hợp d>=2t+1 và s_i =0

Ta biến đổi một trong hai hệ số DCT b’(i,j) và b’(p,q) như sau:

max(|b’(i,j)| , |b’(p,q)|) - (d-INT(0.25*a))

Khi đó giá trị d mới là :

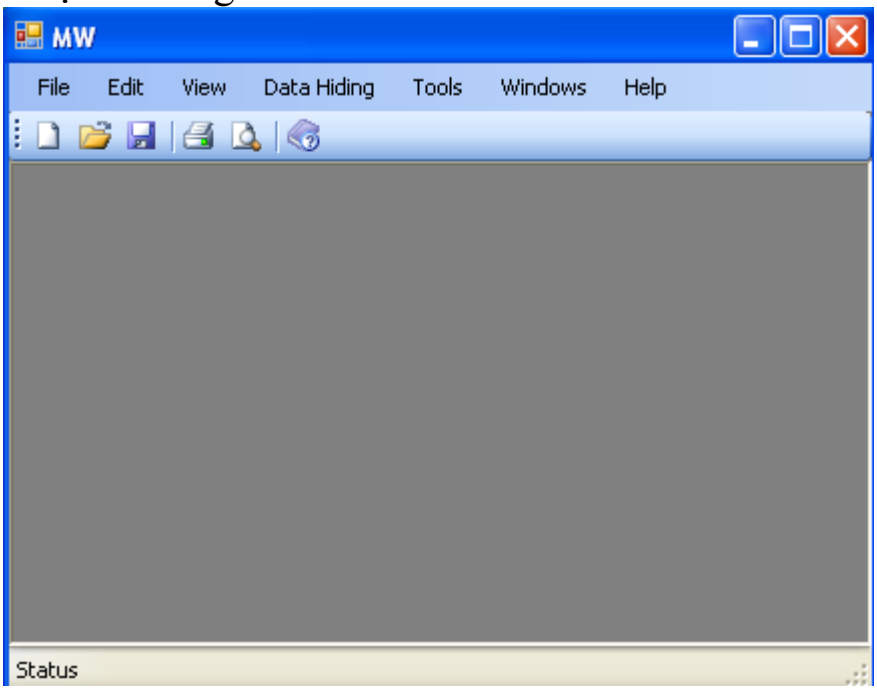
d’ = (||b’(i,j)| -|b’(p,q)| |- (d-INT(0.25*a)) mod a

$\Leftrightarrow$ d’ = (|b’(i,j)| - |b’(p,q)|| mod a)+ (INT(0.25*a)mod a)-(d mod a)

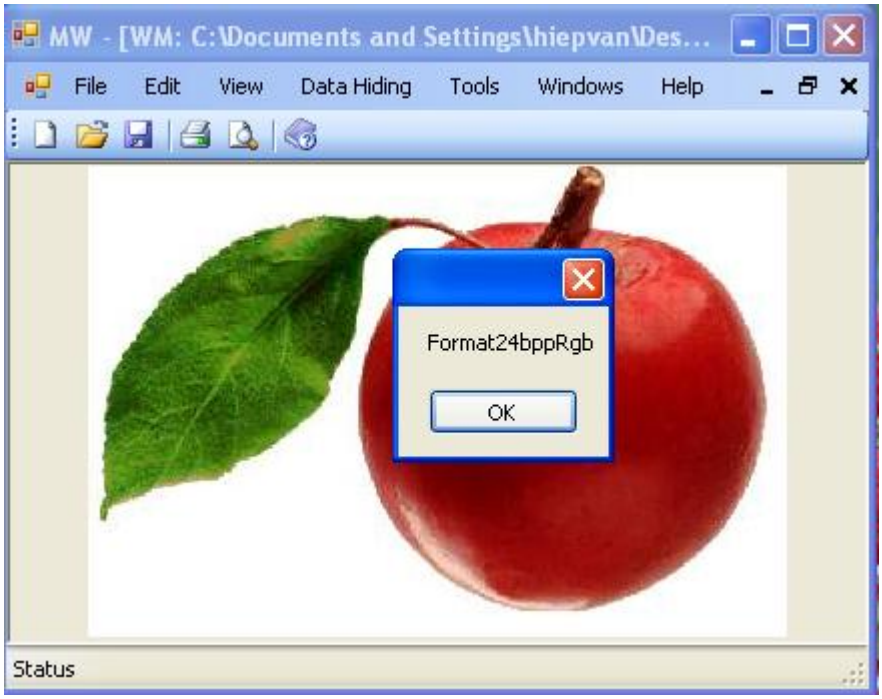
$\Leftrightarrow$ d’ = d-d + 0.25*a= 0.75*a)> 0.5*a=2t+1(dfcm)

Chương 4: **Kết quả thử nghiệm**

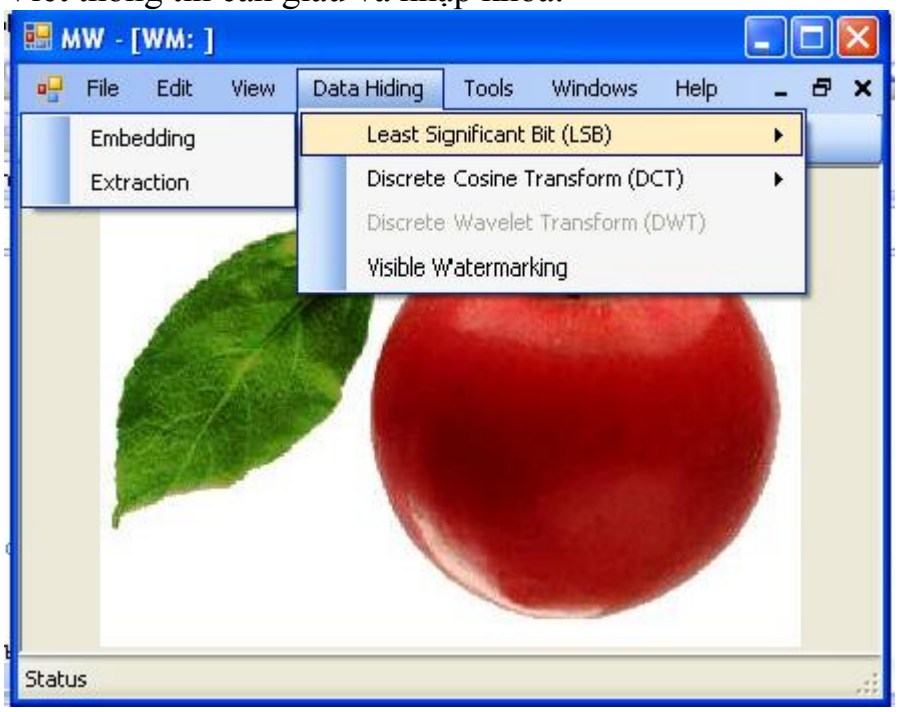
Giao diện chương trình

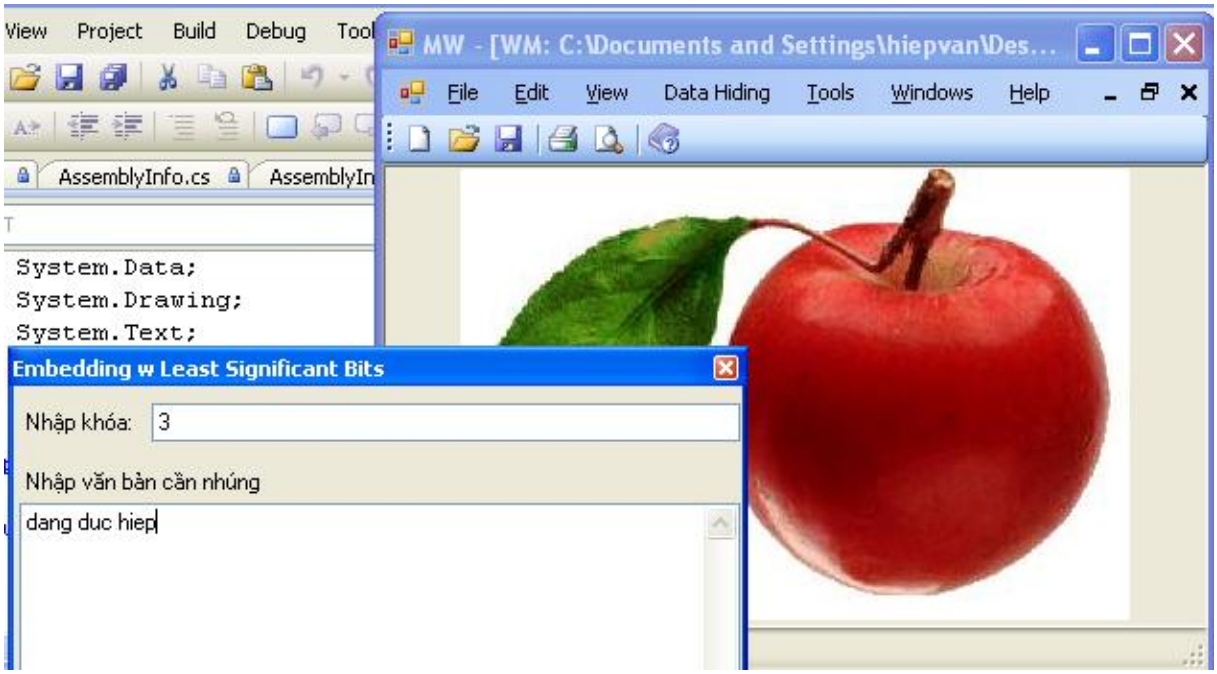


Đọc ảnh dữ liệu đầu vào trong quá trình giấu tin LBS:

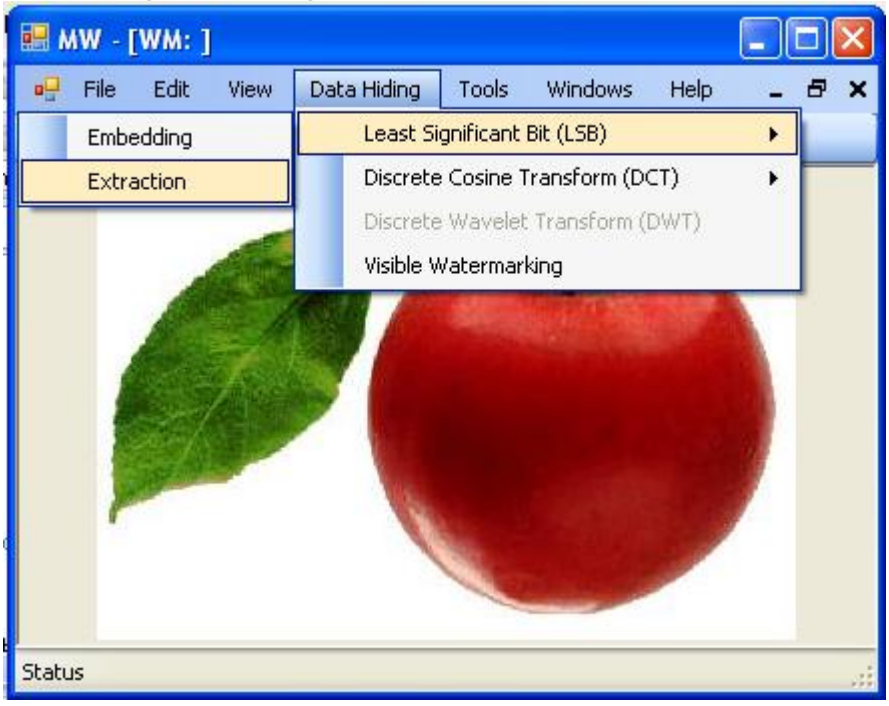


Viết thông tin cần giấu và nhập khóa:

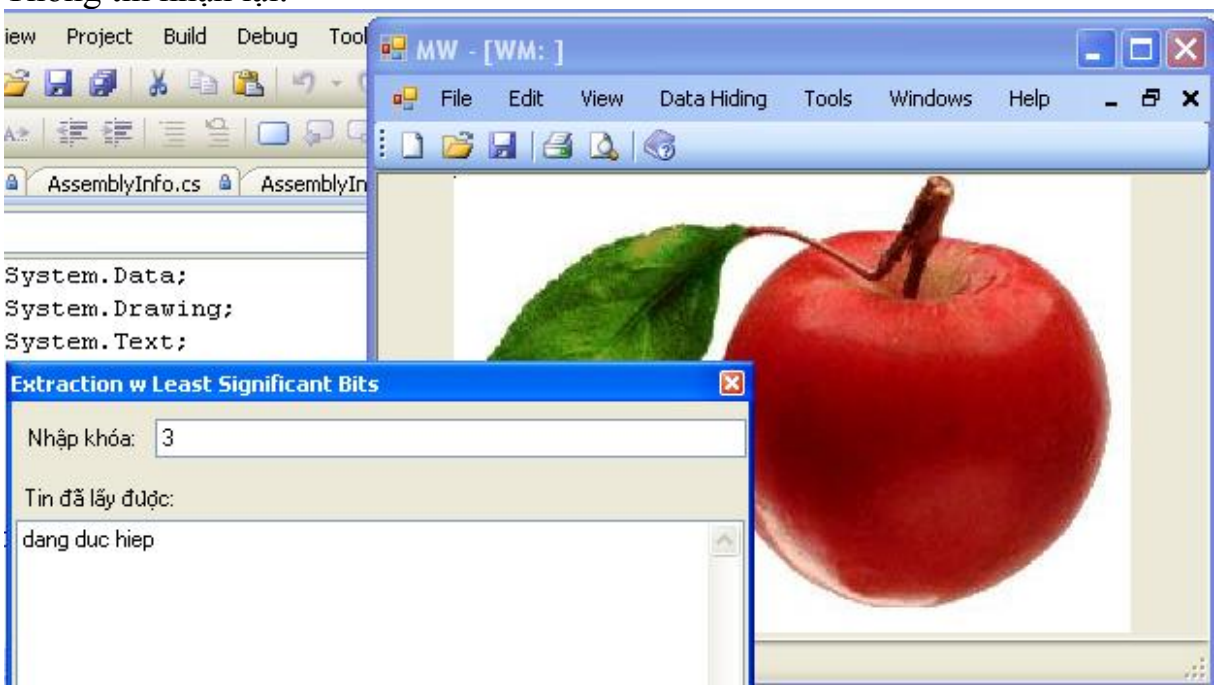




Đọc thông tin được giấu:



Thông tin nhận lại:



CÁC TÀI LIỆU THAM KHẢO

[1] . Nguyễn Xuân Huy, Trần Quốc Dũng, " *Giáo trình giấu tin và thuỷ vân ảnh"*.

[2] .