**MỤC LỤC**

[**DANH MỤC HÌNH VẼ** 2](#_Toc530581948)

[**DANH MỤC BẢNG BIỂU** 3](#_Toc530581949)

[**DANH MỤC VIẾT TẮT** 3](#_Toc530581950)

[**LỜI MỞ ĐẦU** 4](#_Toc530581951)

[**CHƯƠNG I: TỔNG QUAN VỀ THỦY VÂN** 5](#_Toc530581952)

[**1.1 GIẤU TIN TRONG ĐA PHƯƠNG TIỆN** 5](#_Toc530581953)

[***1.1.1*** ***Khái niệm giấu tin*** 5](#_Toc530581954)

[***1.1.2*** ***Mô hình kĩ thuật giấu tin*** 6](#_Toc530581955)

[***1.1.3*** ***Phân loại kĩ thuật giấu tin*** 7](#_Toc530581956)

[***1.1.4*** ***Ứng dụng của kĩ thuật giấu tin*** 9](#_Toc530581957)

[**1.2** **KHÁI NIỆM THỦY VÂN** 10](#_Toc530581958)

[**1.3** **MÔ HÌNH CHUNG CỦA KĨ THUÂT THỦY VÂN** 12](#_Toc530581959)

[**1.4** **PHÂN LOẠI KĨ THUẬT THỦY VÂN** 14](#_Toc530581960)

[***1.4.1*** ***Dựa trên độ bền vững*** 14](#_Toc530581961)

[***1.4.2*** ***Dựa trên hình thức*** 14](#_Toc530581962)

[***1.4.3*** ***Dựa trên tín hiệu số được nhúng*** 14](#_Toc530581963)

[***1.4.4*** ***Dựa trên ứng dụng*** 15](#_Toc530581964)

[***1.4.5*** ***Dựa trên hướng tiếp cận của thủy vân*** 15](#_Toc530581965)

[***1.4.6*** ***Dựa trên khả năng cảm nhận*** 16](#_Toc530581966)

[***1.4.7*** ***Dựa trên mã hóa*** 16](#_Toc530581967)

[**1.5** **ĐÁNH GIÁ CHẤT LƯỢNG ẢNH BẰNG TỈ SỐ TÍN HIỆU CỰC ĐẠI TRÊN NHIỄU (PSNR – Peak Signal-to-Noise)** 16](#_Toc530581968)

[**CHƯƠNG II: MỘT SỐ PHƯƠNG PHÁP THỦY VÂN TRÊN ẢNH SỐ** 17](#_Toc530581969)

[**2.1 TỔNG QUAN VỀ ẢNH SỐ** 17](#_Toc530581970)

[**2.2** **CÁC PHƯƠNG PHÁP THỦY VÂN CHO ẢNH SỐ** 22](#_Toc530581971)

[***2.2.1*** ***Phương pháp Thủy Vân với hướng tiếp cận miền tần số.*** 22](#_Toc530581972)

[*a, Biến đổi Cosine rời rạc* 22](#_Toc530581973)

[*b, Biến đổi Wavelet rời rạc* 23](#_Toc530581974)

[*c, Biến đổi Fourier rời rạc* 24](#_Toc530581975)

[***2.2.2*** ***Phương pháp Thủy Vân với hướng tiệp cận miền không gian.*** 25](#_Toc530581976)

[*a, Thủy vân trực tiếp - Additive Watermarking* 25](#_Toc530581977)

[*b, Kĩ thuật bit ít quan trọng - Least Significant Bit* 25](#_Toc530581978)

[*c, Kĩ thuật mã hóa ánh xạ kết cấu – Texture Mapping Coding Technique* 26](#_Toc530581979)

[*d, Thuật toán chắp vá – Patchwork Algorithm* 26](#_Toc530581980)

[*e, Kĩ thuật tương quan – Correlation Based Technique* 26](#_Toc530581981)

[**2.3** **SO SÁNH CÁC PHƯƠNG PHÁP THỦY VÂN** 26](#_Toc530581982)

[***2.3.1*** ***So sánh các kĩ thuật có hướng tiếp cận miền tần số*** 27](#_Toc530581983)

[***2.3.2*** ***So sánh các kĩ thuật có hướng tiếp cận miền không gian*** 27](#_Toc530581984)

[**CHƯƠNG III: PHƯƠNG PHÁP WATERMARKING CHO ẢNH SỐ SỬ DỤNG KĨ THUẬT BIT CÓ TRỌNG SỐ THẤP** 28](#_Toc530581985)

[**3.1** **BIT CÓ TRỌNG SỐ THẤP (LEAST SIGNIFICANT BIT – LSB)** 28](#_Toc530581986)

[**3.2** **PHƯƠNG PHÁP THỦY VÂN SỬ DỤNG KĨ THUẬT BIT CÓ TRỌNG SỐ THẤP (LSB)** 30](#_Toc530581987)

[***3.2.1*** ***Mô tả phương pháp Thủy Vân sử dụng kĩ thuật LSB*** 30](#_Toc530581988)

[*a,Thuật toán giấu* 30](#_Toc530581989)

[*b, Thuật toán tách* 31](#_Toc530581990)

[***3.3.2*** ***Mô tả phương pháp Thủy Vân sử dụng kĩ thuật LSB và toán tử XOR*** 32](#_Toc530581991)

[*a, Thuật toán giấu* 32](#_Toc530581992)

[*b, Thuật toán tách* 33](#_Toc530581993)

[**3.3** **MÔ PHỎNG PHƯƠNG PHÁP THỦY VÂN SỬ DỤNG KĨ THUẬT BIT CÓ TRỌNG SỐ THẤP** 34](#_Toc530581994)

[***3.3.1*** ***Môi trường cài đặt - Python*** 34](#_Toc530581995)

[***3.3.2*** ***Giao diện chương trình chính*** 35](#_Toc530581996)

[***3.3.3*** ***Giao diện chương trình PSNR*** 41](#_Toc530581997)

[***3.3.4*** ***Thử nghiệm và nhận xét*** 42](#_Toc530581998)

[**KẾT LUẬN** 46](#_Toc530581999)

[**TÀI LIỆU THAM KHẢO** 47](#_Toc530582000)

[**PHỤ LỤC A. CODE PYTHON CHƯƠNG TRÌNH GIẤU TIN XỬ DỤNG KĨ THUẬT LSB** 48](#_Toc530582001)

[**PHỤ LỤC B. CODE PYTHON CHƯƠNG TRÌNH TÍNH TOÁN PSNR** 52](#_Toc530582002)

# DANH MỤC HÌNH VẼ

**CHƯƠNG I: TỔNG QUAN VỀ THỦY VÂN**

[*Hình 1. 1 Mô hình kĩ thuật giấu tin cơ bản 6*](#_Toc530581029)

[*Hình 1. 2 Mô hình quá trình kĩ thuật giấu tin 7*](#_Toc530581030)

[*Hình 1. 3 Phân loại các kỹ thuật giấu tin 9*](#_Toc530581031)

[*Hình 1. 4 Khối mã hóa 13*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581032)

[*Hình 1. 5 Khối giải mã 13*](#_Toc530581033)

**CHƯƠNG II: MỘT SỐ PHƯƠNG PHÁP THỦY VÂN TRÊN ẢNH SỐ**

[*Hình 2. 1 Ảnh đen trắng 18*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581050)

[*Hình 2. 2 Ảnh đa cấp xám 18*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581051)

[*Hình 2. 3 Ảnh màu 19*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581052)

**CHƯƠNG III: PHƯƠNG PHÁP WATERMARKING CHO ẢNH SỐ SỬ DỤNG KĨ THUẬT BIT CÓ TRỌNG SỐ THẤP**

[*Hình 3. 1 Mô tả ba kênh cơ bản của ảnh màu RGB 28*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581056)

[*Hình 3. 2 Biểu diễn RGB của một pixel 29*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581057)

[*Hình 3. 3 Biểu diễn số thập phân bằng số nhị phân 29*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581058)

[*Hình 3. 4 Giao diện chính của chương trình 36*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581059)

[*Hình 3. 5 Giao diện trước khi chạy chương trình 36*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581060)

[*Hình 3. 6 Giao diện đang chạy của chương trình 37*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581061)

[*Hình 3. 7 Giao diện sau khi hoàn thành chương trình 38*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581062)

[*Hình 3. 8 Hình ảnh được nhúng Thủy Vân 38*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581063)

[*Hình 3. 9 Giao diện ở chế độ tách thủy vân 39*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581064)

[*Hình 3. 10 Chương trình tách Thủy vân hoàn thành 40*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581065)

[*Hình 3. 11 Giao diện khi muốn thoát chương trình 40*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581066)

[*Hình 3. 12 Giao diện chương trình nếu bị lỗi 41*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581067)

[*Hình 3. 13 Giao diện chương trình PSNR 41*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581068)

[*Hình 3. 14 Giao diện sau khi chạy xong chương trình 42*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581069)

[*Hình 3. 15 Tập ảnh gốc 42*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581070)

[*Hình 3. 16 Tập ảnh Thủy Vân 43*](file:///C:\Users\Trung\Desktop\BÁO%20CÁO%20TỐT%20NGHIỆP\Báo%20cáo%20đồ%20án\BÁO%20CÁO%20ĐỒ%20ÁN.docx#_Toc530581071)

# DANH MỤC BẢNG BIỂU

**CHƯƠNG I: TỔNG QUAN VỀ THỦY VÂN**

**CHƯƠNG II: MỘT SỐ PHƯƠNG PHÁP THỦY VÂN TRÊN ẢNH SỐ**

[*Bảng 2. 1 Bảng so sánh ưu điểm và nhược điểm của các kĩ thuật có hướng tiếp cận miền tần số 27*](#_Toc530581397)

[*Bảng 2. 2 Bảng so sánh ưu điểm và nhược điểm của các kĩ thuật có hướng tiếp cận miền không gian 27*](#_Toc530581398)

**CHƯƠNG III: PHƯƠNG PHÁP WATERMARKING CHO ẢNH SỐ SỬ DỤNG KĨ THUẬT BIT CÓ TRỌNG SỐ THẤP**

[*Bảng 3. 1 Bảng kết quả đo PSNR trên tập ảnh gốc 43*](#_Toc530581405)

[*Bảng 3. 2 Bảng kết quả đo PSNR trên tập ảnh Thủy Vân 44*](#_Toc530581406)

# DANH MỤC VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **JPEG** | **Joint Photographic Expert Group** | Định dạng nén JPEG |
| ***DCT*** | **Discrete Cosine Transform** | Biến đổi Cosine rời rạc |
| ***DWT*** | **Discrete Wavelet Transform** | Biến đổi Wavelet rời rạc |
| ***DFT*** | **Discrete Fourier Transform** | Biến đổi Fourier rời rạc |
| ***PSNR*** | **Peak Signal – to – Noise** | Tỉ số tín hiệu cực đại trên nhiễu |
| ***IMG*** | **Image** | Định dạng ảnh IMG |
| ***PCX*** | **Personal Computer Exchange** | Định dạng ảnh PCX |
| ***GIF*** | **Graphics Interchanger Format** | Định dạng ảnh GIF |
| ***BMP*** | **Bitmap** | Định dạng ảnh BMP |
| ***LSB*** | **Least Significant Bit** | Bit có trọng số thấp |
| ***RGB*** | **Red – Green – Blue** | Mô hình màu RGB (đỏ - xanh lá – xanh dương) |

# LỜI MỞ ĐẦU

Vấn đề giải quyết quyền sở hữu và bản quyền là vấn đề quan trọng cần giải quyết rộng rãi ở thế kỉ số của chúng ta. Nhu cầu phát triển kĩ thuật bảo vệ dữ liệu kĩ thuật số đang có xu hướng tăng dần. Thủy vân số là kĩ thuật mà ta sẽ nhúng một phần dữ liệu số vào một phần dữ liệu số khác. Thủy vân số được dựa trên việc giác quan con người không thể nhận ra những điểm khác thường nhỏ. Kĩ thuật sẽ nhúng thông tin mật vào các files, và truyền các files đó qua mạng internet, bởi vì chúng về hình thức không khác gì những files thường, chúng có thể tránh khỏi sự can thiệp và các thông tin mật đó sẽ rất khó để bị tấn công.

Phần dữ liệu được nhúng sẽ đóng vai trò như vân tay của người dùng, cho phép bảo vệ bản quyền, xác nhận dữ liệu và theo dõi các bản sao chép bất hợp pháp.

Dựa trên cơ sở lý thuyết và những kiến thức có được qua quá trình tìm hiểu về Thủy vân, em sẽ trình bày chi tiết về kĩ thuật bit có trọng số thấp sử dụng trong Thủy vân cho ảnh số.

Vì vậy, em lựa chọn đề tài “*THỦY VÂN CHO ẢNH SỐ SỬ DỤNG KĨ THUẬT LSB*” nhằm mục đích tìm hiểu, nghiên cứu và thực nghiệm, để từ đó có thể giải quyết một số bài toán bảo vệ quyền sở hữu và bản quyề

# CHƯƠNG I: TỔNG QUAN VỀ THỦY VÂN

# GIẤU TIN TRONG ĐA PHƯƠNG TIỆN

Đa phương tiện (Multimedia) là các tập tin truyền thông hay các thông tin đa phương tiện như: graphic (đồ họa); text (văn bản); photo/image (hình ảnh), video, audio (âm thanh), text (văn bản),….

* + 1. ***Khái niệm giấu tin***

Với những vấn đề em đã đề cập ở trên *Lời Mở Đầu:* ăn cắp bản quyền, thay đổi thông tin, truy cập thông tin trái phép,.. đều tăng dần cùng với sự phát triển của Internet và các đa phương tiện (multimedia); đòi hỏi ta cần phải tìm ra các giải pháp mới, an toàn và bảo mật để có thể khắc phục những vấn đề trên. Một trong các giải pháp được nghiên cứu và áp dụng đó chính là *giấu tin* (Data Hiding).

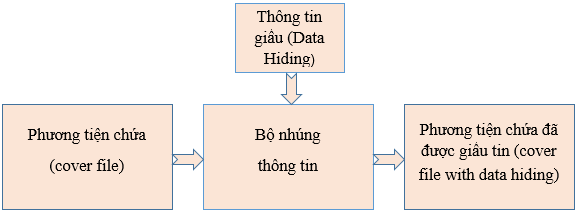
*Giấu tin trong đa phương tiện* là kĩ thuật nhúng (embedding) thông tin số vào một đối tượng dữ liệu số khác. Sự khác biệt giữa mật mã (cryptography) và giấu tin (data hiding) là ở mục đích sử dụng. Ở mật mã, mục đích của mật mã là hạn chế cho phép bên thứ ba có thể bắt được thông tin, nhưng sẽ không thể giải mã được thông tin phía bên trong nếu bắt được thông tin mật đó; còn ở giấu tin, mục đích của kĩ thuật là che giấu, không cho phép bên thứ ba có thể biết được sự tồn tại của thông tin giấu bên trong.

Hiện tại thì giấu tin được chia làm hai nhánh lớn là *giấu tin mật* (Steganography), và *thủy vân* (Watermarking). Phương pháp thủy vân có yêu cầu lưu trữ các mẩu tin ngắn nhưng cần có độ bền vững (roburst) trước các biến đổi thông thường của dữ liệu. Phương pháp giấu tin mật thì yêu cầu che giấu các bản tin đòi hỏi độ bảo mật và dung lượng càng lớn càng tốt.

Kĩ thuật giấu tin đã được nghiên cứu và áp dụng trong nhiều kiểu dữ liệu khác nhau trong đa phương tiện như (text, photo, audio, video,…).

* + 1. ***Mô hình kĩ thuật giấu tin***

Giấu thông tin vào phương tiện chứa và tách lấy thông tin đã giấu là hai quá trình ngược nhau và có thể mô tả qua sơ đồ khối sau:



Hình 1. 1 Mô hình kĩ thuật giấu tin cơ bản

Ở hình 1.1, ta có thể thấy 4 khối tượng trưng có các phần cần xử lý khi áp dụng kĩ thuật giấu tin:

Đầu vào

- Thông tin cần giấu: tùy theo mục đích người sử dụng, nó có thể là mẩu tin (với các tin bí mật) hay là các logo, hình ảnh bản quyền.

- Phương tiện lưu chứa (cover file): các file image, text, audio, video,…

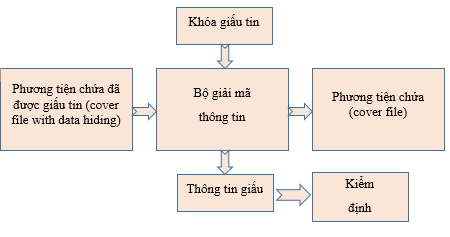
- Bộ nhúng thông tin: là các chương trình mang chức năng thực hiện việc giấu tin.

Đầu ra

- Sản phẩm cuối cùng: phương tiện chứa đã được giấu thông tin.

Tách thông tin diễn ra theo quy trình ngược lại mô hình kĩ thuật giấu tin kể trên. Đầu vào sẽ là phương tiện chứa đã được giấu thông tin. Đầu ra của thông tin sẽ là thông tin đã giấu, ta sẽ quản lý và sử dụng thông tin đó theo những mục đích và yêu cầu khác nhau.

Ở hình 1.2, ta sẽ hình dung ra mô hình quá trình giải mã kĩ thuật giấu tin:



Hình 1. 2 Mô hình quá trình kĩ thuật giấu tin

Sau khi nhận được phương tiện chứa đã được giấu tin, quá trình giải mã được thực hiện qua một bộ giải mã tương ứng, kết hợp cùng với khóa giấu tin, ta sẽ nhận được hai sản phẩm đó là thông tin giấu và phương tiện chứa như ban đầu. Bước tiếp theo thông tin đã giấu sẽ được xử lý kiểm định so sánh với thông tin ban đầu.

* + 1. ***Phân loại kĩ thuật giấu tin***

1. *Phân loại theo phương tiện chứa*

- Giấu tin trong ảnh: chiếm tỉ lệ lớn nhất trong các chương trình ứng dụng hệ thống giấu tin đa phương tiện. Bởi nhu cầu trao đổi thông tin bằng hình ảnh rất lớn, và nhu cầu giấu tin trong ảnh rất lớn với vai trò quan trọng trong các ứng dụng bảo vệ an toàn thông tin như: xác thực, kiểm định sửa đổi, bảo vệ bản quyền tác giả, điều khiển truy cập,…

- Giấu tin trong file âm thanh (audio): có những đặc điểm khác so với giấu tin trong các đối tượng đa phương tiện khác. Do kĩ thuật giấu tin thành công hay không dựa vào hệ thống thính giác, một trong những yêu cầu cơ bản của giấu tin là đảm bảo tính chất giấu tin và không làm ảnh hưởng đến chất lượng dữ liệu gốc.

- Giấu tin vào video: Ý tưởng cơ bản của phương pháp là phân phối thông tin giấu dàn trải theo tần số của dữ liệu gốc. Trong các thuật toán khởi nguồn thì các kĩ thuật cho phép giấu hình ảnh vào video, nhưng gần đây các kĩ thuật đã cho phép giấu cả âm thanh và hình ảnh vào video.

- Giấu tin vào văn bản (text): khó thực hiện hơn do ít các dữ liệu dư thừa, để làm được điều này người ta phải khéo léo khai thác các dư thừa của ngôn ngữ, một cách khác là tận dụng các định dạng văn bản (giấu các thông tin vào khoảng các giữa các từ hoặc các dòng văn bản).

1. *Phân loại theo mục đích sử dụng*

- Giấu thông tin bí mật: đây là ứng dụng phổ biến nhất từ trước đến nay, đối với giấu thông tin mật, người ta quan tâm chủ yếu đến:

+ Độ an toàn của giấu tin – khả năng không bị phát hiện của giấu tin.

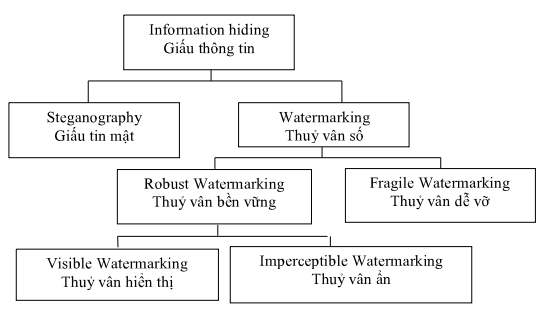
+ Lượng thông tin tối đa có thể giấu trong một phương tiện cụ thể mà vẫn có thể đảm bảo an toàn.

+ Độ bí mật của thông tin trong trường hợp bị phát hiện.

- Giấu thông tin bí mật không quan tâm tới các yêu cầu bền vững của phương tiện chứa.

- Giấu thông tin thủy vân: do yêu cầu bảo vệ bản quyền, xác thực,… nên giấu tin thủy vân có yêu cầu khác với giấu tin bí mật. Yêu cầu đầu tiên là các thủy vân đủ bền vũng trước các tấn công vô hình hay cố ý gỡ bỏ nó. Thêm vào đó các thủy vân phải đảm bảo sự nhận thức của giác quan của con người.

Các kĩ thuật giấu tin mới được phát triển mạnh trong khoảng mười năm trở lại đây nên việc phân loại các kỹ thuật chưa hoàn toàn thống nhất. Sơ đồ phân loại do F.Petitcolas đưa ra năm 1999 được nhiều người chấp nhận.



Hình 1. 3 Phân loại các kỹ thuật giấu tin

Theo sơ đồ, giấu tin được chia thành hai hướng chính là giấu tin mật và thủy vân số. Giấu tin mật quan tâm chủ yếu đến lượng tin có thể giấu, trong từng hướng chính lại chia ra các hương nhỏ hơn, chẳng hạn với thủy vân số thì có thủy vân bền vững và thủy vân dễ vỡ. Thủy vân bền vững có thể đảm bảo thủy vân không bị ảnh hưởng bởi những sự thay đổi của ảnh như dịch chuyển, cắt xén, xoay ảnh. Còn thủy vân dễ vỡ lại không đảm bảo được những cuộc tấn công đó.

* + 1. ***Ứng dụng của kĩ thuật giấu tin***
* **Xác thực thông tin và phát hiện sự thay đổi thông tin (Authentication and Tamper Detection):** Ta sẽ sử dụng một tập tin (sử dụng để giấu thông tin) để xác thực xem dữ liệu đó có phải là bản gốc, và có những tác động như chỉnh sửa, thay đổi không. Tuy nhiên, ta cũng cần phải kiểm soát xem tập tin xác thực này có bị thay đổi hay không. Trong các ứng dụng thực tế, người ta mong muốn tìm được vị trí bị thay đổi cũng như phân biệt được các thay đổi (ví dụ như phân biệt xem một đối tượng đa phương tiện chứa thông tin đã bị thay đổi nội dung hay đơn giản chỉ là bị nén mất dữ liệu). Yêu cầu chung đối với ứng dụng xác thực và phát hiện sự thay đổi thông tin là khả năng giấu thông tin cao và tệp tin được giấu không cần đảm bảo sự bền vững.
* **Bảo vệ bản quyền tác giả (Copyright Protection):** Đây là ứng dụng cơ bản của kỹ thuật giấu tin nói chung và thủy vân số (digital watermarking) nói riêng. Một tệp thông tin mang ý nghĩa kiểm soát bản quyền sở hữu của tác giả sẽ được nhúng vào trong các sản phẩm, tệp thông tin đó chỉ người sở hữu hợp pháp các sản phẩm đó sở hữu, và được dùng làm bằng chứng bản quyền sản phẩm. Ví dụ ta có một sản phẩm đa phương tiện như ảnh, âm thanh, video, và cần được sử dụng, lưu thông thông qua mạng internet. Để đảm bảo rằng các sản phẩm này bị đánh cắp, làm nhái, tái sử dụng trái phép, ta cần phải đưa ra giải pháp chứng minh rằng bản quyền sở hữu là của mình. Giải pháp cơ bản nhất sẽ là “dán nhãn bản quyền” của ta lên sản phẩm đó. Với ứng dụng của giấu tin, ta sẽ “dán nhãn bản quyền” vào sản phẩm mà đối với người sử dụng khác, họ không thể cảm nhận sự thay đổi của sản phẩm kể cả khi “dán nhãn”. Với giấu tin, mẩu tin cần giấu phải có yêu cầu tồn tại bền vững cùng sản phẩm, muốn loại bỏ mẩu tin này, cách duy nhất chỉ có thể là sửa đổi, phá hủy sản phẩm của ta.
* **Vân tay và dán nhãn (Fingerprint and Labelling):** Ứng dụng của giấu tin được sử dụng để nhận diện người gửi hoặc người nhận khi hai người muốn trao đổi thông tin mà muốn xác thực người gửi và nhận có phải chính họ hay không. Ví dụ như các dấu vân tay khác nhau sẽ được nhúng vào bản copy khác nhau của thông tin gốc trước khi chuyển cho nhiều người. Với những ứng dụng này thì yêu cầu là đảm bảo độ an toàn cao cho các mẩu tin tránh sự xóa giấu vết khi trao đổi thông tin.
* **Kiểm soát sao chép (Copy Control):** Các mẩu tin được giấu tin trong trường hợp này được sử dụng để kiểm soát sao chép đối với các thông tin. Các thiết bị phát hiện ra mẩu tin này sẽ được tích hợp sẵn với các hệ thông đọc và ghi. Ví dụ hệ thống quản lí sao chép DVD đã được ứng dụng ở Nhật. Các ứng dụng này cũng yêu cầu mẩu tin có tính bền vững nhất định, và cũng sử dụng phương pháp trích xuất mẩu tin đã giấu mà không cần thông tin gốc.
  1. **KHÁI NIỆM THỦY VÂN**

Dữ liệu số có thể dễ dàng bị sao chép. Điều này rất hữu dụng nhưng cũng mang đến vấn đề đối với những dữ liệu có giá trị, ví dụ, hình ảnh số hoặc nhạc số. Bản sao của dữ liệu số không thể phân biệt và không thể phát hiện bản gốc. Ta không thể phân biệt đâu là bản gốc và đâu là bản sao chép. Các thuật toán ẩn để ngăn cản sự sao chép dữ liệu lại khá dễ dàng bị thay đổi bởi các thuật toán tìm ra thuật toán ngăn cản đó.

Thủy vân theo phân loại phụ thuộc hệ thống thị giác, ta có hai loại: Thủy vân có thể nhìn thầy (visible watermark) và Thủy vân tàng hình (inperceptible watermark).

Thủy vân số cung cấp khả năng nhúng một mẩu tin (message) vào dữ liệu số mà không phá hủy giá trị của nó. Chúng (mẩu tin - messages) có thể coi là một cách định danh của dữ liệu. Các kĩ thuật này có thể sử dụng trên nhiều loại dữ liệu số như hình ảnh, phim ảnh và âm thanh.

Thủy vân số tàng hình cho ta một vài lợi ích hơn so với các kĩ thuật thuật toán ẩn. Tương tự với một kĩ thuật mật mã tốt, chi tiết của một kĩ thuật thủy vân số có thể công khai. Bởi vì các kĩ thuật này có thể tránh khỏi sự thao túng dữ liệu như tìm ra thủy vân và phá hủy nó khỏi dữ liệu cần được đánh dấu.

Thủy vân số tàng hình cung cấp cho người sử dụng đóng dấu dữ liệu của mình một cách bí mật, nó sẽ không phát hiện bằng mắt thường được. Những con dấu có thể nhúng vào dữ liệu để đánh dấu những bức ảnh có giá trị, ví dụ, những con dấu này cho phép người dùng đăng tải hình ảnh để trưng bày, nhưng hợp pháp khi cung cấp một bản quyền nhúng để chặn người khác có thể đăng tải hình ảnh tương tự như vậy.

Vấn đề Thủy Vân và các sự tấn công đến Thủy Vân lại là vấn đề đi liền với nhau. Mục đích của cả hai đều là bảo toàn giá trị của dữ liệu số. Tuy nhiên, mục đích của Thủy Vân là phải đủ sự bền vững để chống khỏi các yếu tố ảnh hưởng và sự tấn công nhưng không ảnh hưởng đến giá trị dữ liệu đang bảo vệ. Còn mặt khác, mục đích của tấn công Thủy Vân là loại bỏ Thủy Vân mà không phá hủy giá trị của dữ liệu được bảo vệ.

Hình ảnh có lợi thế hơn các dữ liệu số khác (ví dụ như xử lý tài liệu văn bản). Nội dung của hình ảnh có thể đánh dấu mà không hiển thị suy hao giá trị hoặc độc lập của một định dạng cụ thể. Ví dụ, 1 hình bitmap có thể nén thành ảnh JPEG. Kết quả là hình ảnh giảm dung lượng nhưng không thể phân biệt với ảnh gốc. Nói chung, một hình ảnh JPEG level (70%) hoặc cao hơn có thể vẫn phù hợp với mắt người thường. Tài nguyên hình ảnh số cũng cho phép đưa các dữ liệu bổ sung vào hình ảnh mà không làm thay đổi giá trị hình ảnh. Thủy vân coi như sẽ “đi lậu vé”. Mẩu tin sẽ được ẩn trong những không gian dư thừa của hình ảnh và nằm dưới phần mắt người không nhìn thấy được.

Ta có thể nói, khái niệm của ẩn giấu dữ liệu và thủy vân có nguồn gốc từ kĩ thuật giấu tin (steganography).

* 1. **MÔ HÌNH CHUNG CỦA KĨ THUÂT THỦY VÂN**

Thông thường, các bước yêu cầu để nhúng một Thủy Vân bao gồm:

* Tạo ra con dấu.
* Nhúng con dấu.
* Tạo ra Thủy Vân số.

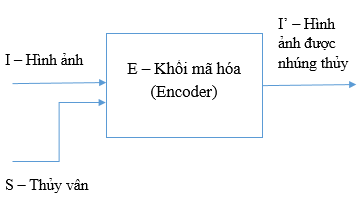
Quá trình mã hóa được mô tả ở Hình 1.4. Ngược lại, quá trình đọc Thủy Vân chỉ cần tệp key và hình ảnh được nhúng Thủy Vân. Quá trình này sẽ được miêu tả ở Hình 1.5.

Trong đó,

I, hình ảnh

S, chữ ký (hay còn được gọi là thủy vân)

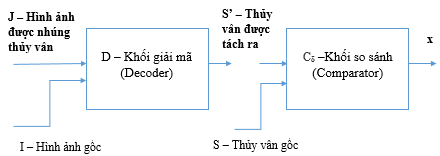
I’, hình ảnh được nhúng thủy vân.

Ở hình 1.4, đầu vào của ta sẽ là 1 hình ảnh I và tệp chữ ký (thủy vân) S, sau đó bằng khối chức năng E sử dụng để mã hóa Thủy Vân, đầu ra sẽ tạo ra một hình ảnh I’ là tệp hình ảnh được nhúng S (thủy vân).

Hình 1. 4 Khối mã hóa

Công thức tổng quát:

|  |  |
| --- | --- |
|  |  |



Hình 1. 5 Khối giải mã

Ở hình 1.5, ta sẽ có hàm chức năng D lấy hình ảnh J (J có thể là một hình ảnh đã có nhúng thủy vân, hoặc chưa có thủy vân trong nó) để xác định rằng có S’ (chữ ký) ở trong hình ảnh đó không. Ở quá trình này, một hình ảnh I có thể được thêm vào (thường nó sẽ là ảnh gốc hoặc ảnh với phiên bản chưa có nhúng thủy vân).

Công thức tổng quát:

|  |  |
| --- | --- |
|  |  |

Thùy vân S’ sẽ được so sánh với thủy vân của người sở hữu bằng một hàm chức năng so sánh:

|  |  |
| --- | --- |
|  |  |

* 1. **PHÂN LOẠI KĨ THUẬT THỦY VÂN**

Thủy vân và kĩ thuật Thủy Vân có thể phân loại thành nhiều phân mục khác nhau dựa trên nhiều yếu tố.

* + 1. ***Dựa trên độ bền vững***
* Fragile Watermarking (Thủy vân yếu): Được sử dụng cho sự bảo vệ tính toàn vẹn của dữ liệu, thủy vân này rất nhạy cảm với sự thay đổi tín hiệu. Qua đó, ta có thể đoán rằng dữ liệu gốc có bị thay đổi hay không.
* Semi Fragile Watermarking (Thủy vân trung bình): Nó có thể tránh được một vài thay đổi không mong muốn như nhiễu.
* Robust Watermarking (Thủy vân bền vững): Được sử dụng chủ yếu để bảo vệ khỏi nhiều loại nhiễu khác nhau với mục đích bảo vệ thủy vân khỏi những thay đổi không mong muốn và có thể dễ dàng phát hiện bởi các phương pháp cụ thể, cung cấp những thông tin xác thực.
  + 1. ***Dựa trên hình thức***
* Visible Watermarking (Thủy vân có thể nhìn thấy): Có thể nhìn thấy được (ví dụ như logo VTV trên kênh truyền hình của VTV,…)
* Invisible Watermarking (Thủy vân không thể nhìn thấy): Bằng kĩ thuật này, ta có thể giấu những thông tin mật vào phương tiện số như hình ảnh, mà mắt thường không thể thấy được. Nó chỉ có thể được phát hiện bằng một quá trình cụ thể.
  + 1. ***Dựa trên tín hiệu số được nhúng***
* Image Watermarking (Thủy vân cho ảnh số): Được sử dụng để nhúng tin vào dữ liệu gốc là hình ảnh số.
* Video Watermarking (Thủy vân cho video): Được nhúng vào dữ liệu của video thông thường hoặc video đang streaming. Đối với video streaming, ta yêu cầu kĩ thuật trích xuất thời gian thực.
* Audio Watermarking (Thủy vân cho âm thanh): Sử dụng cho hệ thống âm thanh như MP3.
  + 1. ***Dựa trên ứng dụng***
* Xác thực dữ liệu và tích hợp thủy vân (Data Authentication and Intergrity Watermarking): Kiểm soát dữ liệu và xác thực dữ liệu khi cần thiết.
* Bảo vệ bản quyền (Copyright Protection Watermarking): Tác giả sở hữu có thể kiểm tra lại con dấu, chữ ký, hay bất kì thông tin gì đã được nhúng trong sản phẩm và có thể truy xuất để chứng minh bản quyền sở hữu của mình.
* Thủy vân chống sao chép (Anti-Counterfelting Watermarking): Được sử dụng với những tài liệu dạng văn bản, mặc dù in hoặc scan chúng, ta vẫn có thể tìm và phát hiện ra loại thủy vân này.
  + 1. ***Dựa trên hướng tiếp cận của thủy vân***
* Hướng tiếp cận miền không gian (Spatial Domain Watermarking): Hướng tiếp cận này cơ bản sẽ chỉnh sửa thay đổi pixel của dữ liệu gốc. Một vài thuật toán sử dụng hướng tiếp cận này là LSB, SSM. Độ phức tạp của các thuật toán này đều thấp, và ứng dụng của chúng thường được dùng là xác thực.
* Hướng tiếp cận miền tần số (Frequency Domain Watermarking): Trong kĩ thuật này thì một số tần số nhất định sẽ được thay đổi để có thể mang khả năng giấu tin. Một số kĩ thuật hướng tiếp cận tần số là DCT, DWT hoặc DFT. Độ phức tạp của các thuật toán này cao, và thường được sử dụng ứng dụng bảo vệ bản quyền.
  + 1. ***Dựa trên khả năng cảm nhận***
* Thủy vân có thể nhìn thấy (Visual Watermarking): Thủy vân có độ bền vững cao những ứng dụng khá hạn chế.
* Thủy vân bán vô hình (Semi-blind Watermarking): Kĩ thuật không cần đa phương tiện gốc để nhận biết và trích xuất.
* Thủy vân vô hình (Blind Watermarking): Yêu cầu kĩ thuật thủy vân cao và không cần đa phương tiện gốc.
  + 1. ***Dựa trên mã hóa***
* Thủy vân sử dụng mã hóa bất đối xứng ( Asymemtric Watermarking)
* Thủy vân sử dụng mã hóa đối xứng (Symmetric Watermarking)
  1. **ĐÁNH GIÁ CHẤT LƯỢNG ẢNH BẰNG TỈ SỐ TÍN HIỆU CỰC ĐẠI TRÊN NHIỄU (PSNR – Peak Signal-to-Noise)**

Tỉ số tín hiệu cực đại trên nhiễu (PSNR – Peak Signal-to-Noise) là một thuật ngữ dùng để tính tỉ lệ giữa các giá trị năng lượng tối đa của một tín hiệu và năng lượng nhiễu ảnh hưởng đến tín hiệu đó. Ta thường dùng tỉ số này để đánh giá độ nhiễu của ảnh trước và sau khi giấu tin. Đơn vị của tỉ số là logarit decibel vì tỉ số có độ biến thiên rộng. Giá trị PSNR được coi là tốt ở vào khoảng 30 – 50dB, tín hiệu có giá trị dưới 20dB là tín hiệu không thể chấp nhận được. PSNR cũng được dùng rộng rãi trong kỹ thuật đánh giá chất lượng hình ảnh và video sau khi bị biến đổi.

Trong đồ án này, em sẽ định nghĩa thông số PSNR thông qua sai số toàn phương trung bình (MSE – Mean Squared Error) được dùng cho ảnh hai chiều có kích thước m\*n, với I và K là ảnh gốc và ảnh được khôi phục tương ứng:

|  |  |
| --- | --- |
|  |  |

PSNR sẽ được định nghĩa như sau:

|  |  |
| --- | --- |
|  |  |

Ở đây MAXI là giá trị tối đa của pixel trên ảnh. Khi các pixels được biểu diễn 8 bits thì giá trị của nó là 255. Với trường hợp tổng quát, khi tín hiểu được biểu diễn bởi *K* bits trên một đơn vị lấy mẫu, MAXI của tín hiệu là 2*K*-1. Với ảnh màu có 3 giá trị RGB trên 1 pixel, các tính toán của PSNR tương tự ngoại trừ việc tính MSE là tổng của 3 giá trị (tính trên 3 kênh màu) chia cho kích thước của ảnh và chia cho 3.

# CHƯƠNG II: MỘT SỐ PHƯƠNG PHÁP THỦY VÂN TRÊN ẢNH SỐ

# TỔNG QUAN VỀ ẢNH SỐ

* + 1. ***Khái niệm ảnh số***

Ảnh số là hình ảnh được xử lý và lưu trữ trên máy tính. Ảnh số là ma trận 2 chiều, được coi là những thành tố của bức ảnh và thường được biết dưới tên gọi là pixels. Các pixel này sẽ bao gồm các giá trị chứa trong file ảnh để định ra mầu sắc, độ sáng tối của từng pixel trong mạng lưới đó - ảnh số được hình thành. Việc kiểm soát, định ra địa chỉ theo mạng lưới như trên được gọi là *bit mapping*và ảnh số còn được gọi là ảnh *bit-maps (BMP).*

Các thuộc tính tiêu biểu của một tập tin ảnh BMP là:

* Số bit trên mỗi điểm ảnh (bit per pixel), thường được kĩ hiệu là n. Vậy một ảnh BMP n-bit sẽ có 2n màu. Giá trị màu tiêu biểu của n là 1 (đối với ảnh đen trắng), 4 (đối với ảnh 16 màu), 8 (ảnh 256 màu), 16 (ảnh 65536 màu) và 24 (ảnh 16 triệu màu).
* Chiều cao (height), chiều rộng (width) và điểm ảnh (pixel).

Đặc điểm nổi bật nhất của định dạng BMP là tập tin hình ảnh thường không được nén bằng bất kì 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.

* + 1. ***Các dạng ảnh số***

Ảnh bitmap được chia thành ba dạng: ảnh nhị phân (đen trắng), ảnh đa mức xám (Gray Scale) và ảnh màu.

* Ảnh đen trắng (Black and White Image): có thể được gọi là ảnh nhị phân (Binary Image): giá trị mỗi điểm ảnh là 0 và 1, nghĩa là trắng hoặc đen. Trong thực tế khi xử lý trên máy tính thì người ta thường sẽ dùng ảnh đa mức xám, để biểu thị ảnh nhị phân, ảnh đen trắng sẽ trả về hai giá trị là 0 và 255.

Hình 2. 1 Ảnh đen trắng

* Ảnh đa mức xám (Gray Scale Image): giá trị mỗi điểm ảnh nằm trong khoảng giá trị từ 0 đến 255, nghĩa là ta cần 8 bits (1 byte) để biểu diễn mỗi điểm ảnh này.

Hình 2. 2 Ảnh đa cấp xám

* Ảnh màu (Color Image): mỗi điểm ảnh có giá trị gồm 3 màu đỏ (R – red); xanh lục (G – green) và xanh dương (B – blue). Mỗi màu có giá trị từ 0 đến 255, nghĩa là mỗi điểm ảnh thường sẽ cần 24 bits (3 byte) để biểu diễn. Tuy nhiên số lượng màu thực tế của một ảnh thường khá nhỏ, và tùy theo chất lượng ảnh mà quyết định số bit để biểu diễn cho mỗi màu là 24 bits hoặc 32 bits.

Hình 2. 3 Ảnh màu

* + 1. ***Định dạng ảnh thông dụng***

Sau khi được xử lý số hóa hình ảnh, ta sẽ có nhiều loại định dạng ảnh khác nhau, phụ thuộc vào kĩ thuật số hóa ảnh. Dưới đây em xin giới thiệu một số định dạng thông thường.

* 1. *Định dạng ảnh IMG (Image)*

Ảnh IMG là ảnh đen trắng, mỗi điểm ảnh được thể hiện bởi 1 bit. Toàn bộ ảnh chỉ bao gồm các điểm sáng và tối, tương ứng với giá trị 0 và 1.

Tỉ lệ nén của kiểu định dạng này khá cao. Ảnh IMG nén theo từng dòng, mỗi dòng bao gồm các gói (pack). Các dòng giống nhau được nén thành một gói.

* 1. *Định dạng ảnh PCX (Personal Computer Exchange)*

Định dạng ảnh PCX là một trong những định dạng cổ điển nhất. Nó sử dụng phương pháp mã hóa loạt dài RLC để nén dữ liệu ảnh. Quá trình nén và giải nén cũng nằm trên từng dòng ảnh. Thực tế, phương pháp giải nén PCX hiệu quả hơn so với kiểu IMG.

Định dạng ảnh PCX thường được dùng để lưu trữ ảnh vì thao tác đơn giản, cho phép nén và giải nén nhanh. Tuy nhiên vì cấu trúc cố định, nên trong một số trường hợp nó làm tăng kích thước lưu trữ.

* 1. *Định dạng ảnh GIF (Graphics Interchanger Format)*

Định dạng ảnh GIF do Computer Incorporated (Mỹ) đề xuất lần đầu tiên vào năm 1990. Với định dạng GIF, chỉ số màu tăng thì ưu thế của định dạng càng nổi trội. Những ưu thế này có được do GIF sử dụng thuật toán LZW (Lampel – Ziv – Welch) (dựa vào sự lặp lại của một nhóm điểm, tìm kiếm tần suất lặp lại của chúng và thay thế bằng từ mã tương ứng mỗi khi gặp lại nhóm điểm đó. Dạng GIF cho chất lượng ổn, độ phân giải tốt, cho phép hiển thị trên hầu hết các phần cứng đồ họa.

* 1. *Định dạng ảnh BMP (Bitmap)*

Ảnh BMP (Bitmap) được phát triển bởi Microsoft Corporation, được lưu trữ dưới dạng độc lập thiết bị, cho phép Windows hiển thị dữ liệu không phụ thuộc vào khung màu chỉ định trên bất kì phần cứng nào. Tên file mở rộng mặc định của một file ảnh Bitmap là \*.bmp . Quy ước màu đen trắng tương ứng với các giá trí 0 và 1. Ảnh BMP được sử dụng trên Microsoft Windows và các ứng dụng chạy trên Windows

Có nhiều định dạng ảnh thuộc kiểu bitmap như BMP, PCX, TIFF, GIF, JPEG, TGA, PNG, PCD,… Mỗi file ảnh BMP gồm bốn phần:

* Bitmap Header (14 bytes): giúp nhận dạng tập tin bitmap.
* Bitmap Information (40 bytes): chứa một số thông tin chi tiết giúp hiển thị hình ảnh.
* Palette màu (4\*x bytes): x là số màu của ảnh, định nghĩa các màu sẽ được sử dụng trong ảnh.
* Bitmap Data: chứa dữ liệu ảnh.

Đặc điểm nổi bật của định dạng BMP là tập tin ảnh hưởng thường không được nén bằng bất kì 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 lớn, gấp nhiều lần so với các định dạng ảnh được nén (ví dụ như GIF, JPEG hay PNG).

* + 1. ***Định dạng ảnh JPEG (Joint Photographic Expert Group)***

Một nhóm các nhà nghiên cứu đã phát minh ra định dạng này, để hiển thị hình ảnh đầy đủ hơn (full-color) cho định dạng di động, mà kích thước file lại nhỏ hơn. Giống như ảnh GIF, JPEG cũng được sử dụng nhiều trên Web.

Lợi ích của JPEG hơn GIF là nó có thể hiển thị hình ảnh với màu chính xác (true-color) (có thể lên đến 16 triệu màu). Điều đó cho phép JPEG được sử dụng tốt nhất cho hình ảnh chụp và hình ảnh minh họa có số lượng màu lớn.

Nhược điểm chính của định dạng JPEG là chúng được nén bằng thuật toán có tổn hao (lossy compresion). Điều này có nghĩa rằng hình ảnh sẽ 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. Tuy nhiên với các hình ảnh chất lượng màu cao (high-color) như hình ảnh chụp, thì điểu này sẽ không ảnh hưởng gì.

Ảnh JPEG không thể làm trong suốt hoặc chuyển động, trong trường hợp này, ta sẽ sử dụng định dạng GIF hoặc PNG để tạo độ trong suốt.

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 đôi kích thước file JPEG được gọi là chất lượng (quality), và thường có giá trị 0 đến 100%. Trong đó 0% thì chất lượng là thấp nhất (kích thước file là nhỏ nhất) và 100% là chất lượng cao nhất (với kích thước file sẽ là lớn nhất). 0% chất lượng JPEG sẽ nhìn rất mờ khi so sánh với ảnh gốc. Còn 100% chất lượng JPEG thường sẽ không phân biết được so với ảnh gốc.

* 1. **CÁC PHƯƠNG PHÁP THỦY VÂN CHO ẢNH SỐ**

Thủy vân số được định nghĩa là quá trình nhúng thủy vân vào các phương tiện số (ví dụ như âm thanh, video, hình ảnh,…) sử dụng một thuật toán thích hợp. Thủy vân có thể là giải pháp cho nhiều mục đích như phát hiện sự sửa đổi, xác thực dữ liệu, bảo mật, bảo vệ bản quyền. Thuật toán sử dụng cho thủy vân đóng vai trò quan trọng (chúng phải vừa thích hợp, bền vững và đảm bảo những yêu cầu của ứng dụng sử dụng chúng). Bởi vì nếu thuật toán bền vững, hiệu quả và hiệu suất, thủy vân sẽ khó trích xuất hơn. Người có thể trích xuất được thủy vân sẽ là người biết rõ thuật toán sử dụng để nhúng thủy vân nhất. Có khá nhiều thuật toán đang được sử dụng để giấu tin, trong phần này, em sẽ phân loại chúng thành hai loại theo hướng tiếp cận của thủy vân: hướng tiếp cận theo miền tần số và hướng tiếp cận theo miền không gian.

* + 1. ***Phương pháp Thủy Vân với hướng tiếp cận miền tần số.***

Thủy vân với hướng tiếp cận miền tần số sẽ được nhúng vào hệ số quang phổ của hình ảnh. Các thuật toán được sử dụng ở miền tần số là Biến đổi Cosine rời rạc (Discrete Cosine Transform – DCT); Biến đổi Fourier rời rạc (Discrete Fourier Transform – DFT); Biến đổi Wavelet rời rạc (Discrete Wavelet Transform – DWT), thủy vân với hướng tiếp cận miền tần số được áp dụng rộng rãi bởi vì các đặc điểm của hệ thống thị giác con người (Human Visual System - HVS) được nắm bắt hiệu quả bởi hệ số quang phổ hơn. Dưới đây em xin giới thiệu một vài thuật toán chính.

### *a, Biến đổi Cosine rời rạc*

Biến đổi cosine rời rạc (Discrete Cosine Transform – DCT) biểu diễn dữ liệu bằng các khoảng cách tần số chứ không phải bằng biên độ tần số. Kĩ thuật thủy vân sử dụng biến đổi cosine rời rạc được cho là bền vững hơn so với kĩ thuật với hướng tiếp cận miền không gian. Những thuật toán theo hướng tần số đều rất bền vững với những kĩ thuật xử lý hình ảnh số như lọc băng thấp (low pass filter), tinh chỉnh độ sáng (brightness) và điều chỉnh tương phản (contrast). Chúng khá tốn kém và khó để triển khai thực hiện. Cùng lúc đó thì chúng khá yếu đối với những biến đổi hình học như xoay (rotation), chỉnh kích cỡ (scaling) và cắt xén (cropping). Thủy vân sử dụng biến đổi cosine rời rạc có thể phân loại thành thủy vân DCT toàn cục (Global DCT Watermarking) và thủy vân DCT theo khối (Block based DCT watermarking). Một trong những thuật toán DCT đầu tiên được Cox et al. (1997) sử dụng thủy vân DCT toàn cục sẽ nhúng một thủy vân bền vững vào trong vùng không nhận thức của hệ thống thị giác con người.

Sau đây là những bước thực hiện của thuật toán thủy vân DCT theo khối:

1. Hình ảnh sẽ được phân mảnh thành các khối không chồng nhau với kích cỡ 8x8.
2. Áp dụng thuật toán DCT thuận (Forward DCT) với những khối ảnh 8x8 này.
3. Áp dụng lựa chọn khối theo tiêu chuẩn.
4. Áp dụng lựa chọn hệ số theo tiêu chuẩn.
5. Nhúng thủy vân bằng điều chỉnh hệ số ở trên.
6. Áp dụng thuật toán DCT nghịch (Inverse DCT) trên từng khối 8x8.

*b, Biến đổi Wavelet rời rạc*

Đây là kĩ thuật hiện đại, được sử dụng phổ biến trong xử lý tín hiệu số, nén ảnh số, thủy vân,… Biến đổi dựa trên những sự thay đổi của bước sóng (wave) nhỏ, gọi là wavelet, trong nhiều tần số và theo một thời gian giới hạn. Ý tưởng cơ bản của DWT trong xử lý ảnh là đa vi phân hình ảnh theo từng không gian và tần số độc lập.

*Đặc điểm của DWT*:

1. *Biên đổi Wavelet biên soạn lại hình ảnh thành ba hướng không gian: trục tung, trục hoành và đường chéo.*
2. *Độ lớn hệ số DWT sẽ lớn hơn ở vùng thấp nhất (LL) ở mỗi cấp bậc của việc biên soạn và nhỏ hơn ở các vùng còn lại (HH, LH, và HL).*
3. *Thủy vân với mức phân giải thấp thì hiệu quả hơn bởi vì sẽ được nhúng vào ít dải tần số hơn.*
4. *Hệ số wavelet sẽ mang lại hiệu suất cao nếu độ lớn của nó lớn.*

*Ưu điểm của DWT so với DCT*: DWT hiểu rõ hệ thống thị giác con người hơn so với DCT. DWT cung cấp cả hướng tiếp cận không gian và trải phổ theo tần số. Hình ảnh sử dụng thuật toán DWT sẽ có thể sử dụng cả ở độ phân giải thấp và cao với nhiều cấp độ phân giải. Mắt người cũng khó nhận ra dấu vết sử dụng thuật toán hơn so với DCT.

*Nhược điểm của DWT so với DCT*: Tính toán phức tạp hơn. Theo Fieg (1990), ông đã chỉ ra DCT chỉ cần tính 54 phép tính, còn DWT sẽ cần ít nhất 1 phép tính với mỗi hệ số, DWT sẽ phụ thuộc vào độ dài của filter sử dụng.

### *c, Biến đổi Fourier rời rạc*

Biến đổi Fourier rời rạc (Discrete Fourier Transform) có sự bền vững đối với những sự thay đổi như chỉnh sửa kích cỡ (scaling), xoay (rotation), cắt xén (cropping),… Nó biến đổi một hàm liên tục thành các thành phần tần số

*Đặc điểm của DFT:*

1. *DFT có tính chất dịch chuyển bất biến. Các phân vùng của hình ảnh ảnh hưởng đến các pha của hình ảnh những không ảnh hưởng biên độ của hình ảnh.*
2. *DFT của một hình ảnh sẽ thường sử dụng số phức (kết quả của pha và biên độ của hình ảnh).*
3. *Thành phần trung tâm của DFT là thành phần mạnh nhất vì nó chưa tần số thấp nhất.*
4. *Thay đổi kích cỡ hình ảnh sẽ được biểu diễn bởi biên độ của tín hiệu trích xuất và phát hiện bởi sự tương quan của các hệ số.*
5. *Xoay ảnh sẽ được biểu diễn ở các pha của tín hiệu trích xuất và có thể phát hiện bằng khảo sát.*

*Ưu điểm của DFT so với DWT và DCT:* DFT chống lại được các sự tác động hình học như đổi kích cỡ, xoay và cắt xén ảnh, nên kĩ thuật này sẽ được sử dụng khi các kĩ thuật theo miền không gian, DWT và DFT không khắc phục được.

* + 1. ***Phương pháp Thủy Vân với hướng tiệp cận miền không gian.***

Thuật toán thủy vân với hướng tiếp cận miền không gian sẽ xử lý trực tiếp dữ liệu của hình ảnh gốc. Nó có thể áp dụng vào sự phân chia của màu trong từng pixel. Miền không gian sẽ chi phối sự thể hiện của hình. Những thuật toán này dựa trên việc nhúng trực tiếp thủy vân vào trong từng pixel của hình ảnh. Một vài thuật toán em xin được miêu tả dưới đây.

### *a, Thủy vân trực tiếp - Additive Watermarking*

Đây là phương pháp trực tiếp được sử dụng trong kĩ thuật thủy vân có hướng tiếp cận miền không gian để nhúng thủy vân vào đa phương tiện (chủ yếu là hình ảnh). Trong phương pháp này ta sẽ thêm các khuôn mẫu nhiễu ngẫu nhiên pseudo (Pseudo Random Noise Pattern) vào trong các giá trị của pixel. Các tín hiệu nhiễu này sẽ các số nguyên (-1, 0, 1) hoặc các số thực. Để đảm bảo ta có thể trích xuất thủy vân, ta sẽ cần có một key, đó sẽ là căn cứ để ta tìm được các pixel mà thủy vân ngẫu nhiên được nhúng vào những pixel được chọn ngẫu nhiên.

### *b, Kĩ thuật bit ít quan trọng - Least Significant Bit*

Trong kĩ thuật này, ta sẽ nhúng thủy vân vào thành phần bit ít quan trọng nhất của pixel, phương pháp này dễ triển khai nhưng không bền vững so với các cuộc tấn công. Điều đó có nghĩa là thủy vân dễ dàng bị phá hủy. Cách hoạt động của kĩ thuật là ta sẽ đưa các bit của thủy vân vào những thành phần bit ít quan trọng nhất trong từng pixel. Thủy vân sẽ vô cùng nhạy cảm với nhiều và những tín hiệu xử lý thông thường và khó sử dụng vào những ứng dụng thực tế.

Kĩ thuật dựa trên kĩ thuật tương quan:

Trong phương pháp này, một khuôn mẫu chuỗi ngẫu nhiên (Pseudorandom Noise – PN) được kí hiệu là W(x,y) sẽ được thêm vào hình ảnh I(x,y):

Trong đó, K biểu diễn cho hệ số gain, Iw biểu diễn cho hình ảnh đã được nhúng thủy vân ở vị trí x, y và I biểu diễn cho hình ảnh phủ bên ngoài (cover image). Chất lượng hình ảnh có thể tăng hoặc giảm tùy thuộc vào hệ số gain. Sự khác nhau giữa thuật toán bit ít quan trọng LSB và thuật toán tương quan sẽ được miêu tả ở dưới.

### *c, Kĩ thuật mã hóa ánh xạ kết cấu – Texture Mapping Coding Technique*

Kĩ thuật này chỉ hiệu quả với những hình ảnh có các kết cấu. Phương pháp của kĩ thuật này là ta sẽ giấu tin vào trong các phân lớp của hình ảnh đó. Thuật toán này chỉ phù hợp với những khu vực có số lượng lớn các kết cấu độc nhất. Phương pháp này sẽ giấu tin vào các hình mẫu kết cấu ngẫu nhiên liên tục của hình ảnh.

### *d, Thuật toán chắp vá – Patchwork Algorithm*

Thuật toán này là kĩ thuật giấu tin phát triển bởi Bender et al và xuất bản trên tạp chí IBM Systems Journal, 1996. Thuật toán được dựa trên mã pseudo ngẫu nhiên, mô hình thống kê. Thuật toán sẽ đưa thủy vân vào một số điểm thống kê cụ thể dựa vào phân phối thường (phân phối Gaussian). Nó sẽ lựa chọn hai miếng vá (patches) với một dữ liệu hình ảnh sẽ được làm sáng lên và một dữ liệu sẽ được làm tối đi theo thứ tự.

### *e, Kĩ thuật tương quan – Correlation Based Technique*

Trong phương pháp này, một khuôn mẫu chuỗi ngẫu nhiên (Pseudorandom Noise – PN) được kí hiệu là W(x,y) sẽ được thêm vào hình ảnh I(x,y):

Trong đó, K biểu diễn cho hệ số gain, Iw biểu diễn cho hình ảnh đã được nhúng thủy vân ở vị trí x, y và I biểu diễn cho hình ảnh phủ bên ngoài (cover image). Chất lượng hình ảnh có thể tăng hoặc giảm tùy thuộc vào hệ số gain.

* 1. **SO SÁNH CÁC PHƯƠNG PHÁP THỦY VÂN**

## ***So sánh các kĩ thuật có hướng tiếp cận miền tần số***

Bảng 2. 1 Bảng so sánh ưu điểm và nhược điểm của các kĩ thuật có hướng tiếp cận miền tần số

| Thuật toán | Ưu điểm | Nhược điểm |
| --- | --- | --- |
| DCT | - Bền vững đối với những quá trình xử lý số  - Thủy vân không thể bị loại bỏ bởi bất cứ cuộc tấn công nào. Thủy vân nằm hệ số tần số trung. | - Phần tử có tần số cao hơn sẽ có khả năng bị ngăn chặn trong quá trình lượng tử.  - Các khối DCT loại bỏ tính chất bất biến của hệ thống.  - Dễ bị ảnh hưởng bởi cắt xén và thay đổi kích cỡ. |
| DWT | - Tỉ lệ nén cao hơn tạo sự khó nhận thức ở hệ thống thị giác của con người.  - Cho phép xử lý ở cả trên miền thời gian và tần số - không gian. | - Chưa tối ưu vì cần nhiều phép tính toán.  - Các phép tính toán khá phức tạp.  - Thời gian xử lý cần nhiều thời gian hơn.  - Có thể sẽ có nhiễu. |
| DFT | - Sử dụng để hồi phục từ những biến đổi hình học. | - Độ phức tạp cao  - Cần nhiều phép tính toán. |

* + 1. ***So sánh các kĩ thuật có hướng tiếp cận miền không gian***

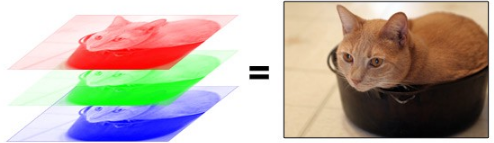
Bảng 2. 2 Bảng so sánh ưu điểm và nhược điểm của các kĩ thuật có hướng tiếp cận miền không gian

| Thuật toán | Ưu điểm | Nhược điểm |
| --- | --- | --- |
| LSB | - Ít ảnh hưởng tới chất lượng ảnh.  - Dễ dàng thực hiện và hiểu.  - Dễ đánh lừa nhận thức của hệ thống thị giác. | - Nhạy cảm với nhiễu.  - Ảnh hưởng bởi cắt xén và thay đổi kích cỡ.  - Kém bền vững với những cuộc tấn công. |
| Tương quan | - Tăng sự bền vững của thuật toán bằng cách tăng hệ số gain. | - Tăng hệ số gain sẽ làm giảm chất lượng ảnh. |
| Chắp vá | - Tăng sự bền vững. | - Ít thông tin có thể giấu hơn |
| Mã hóa ánh xạ kết cấu | - Giấu tin vào những mẫu hình kết cấu ngẫu nhiên liên tục của hình ảnh. | - Thuật toán chỉ phù hợp với những vùng có số lượng lớn kết cấu hình ảnh độc nhất. |

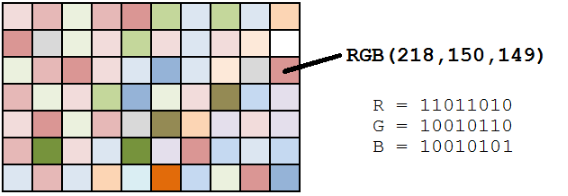
# CHƯƠNG III: PHƯƠNG PHÁP WATERMARKING CHO ẢNH SỐ SỬ DỤNG KĨ THUẬT BIT CÓ TRỌNG SỐ THẤP

* 1. **BIT CÓ TRỌNG SỐ THẤP (LEAST SIGNIFICANT BIT – LSB)**

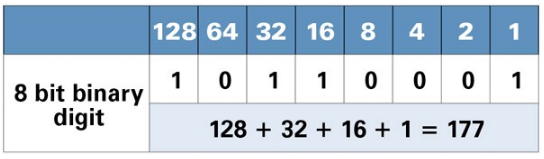
Như ta đã biết ở trên, pixel là đơn vị nhỏ nhất của một hình ảnh. Trong ảnh bitmap đen trắng thì pixel được biểu thị bởi 1 bit giá trị bit 0 và 255 ( tương ứng 0 và 1), với ảnh đa cấp xám (grayscale) thì pixel được biểu thị bởi 8 bit giá trị từ 0 đến 255 (tương ứng với các cấp xám). Còn trong ảnh màu, một pixel biểu thị bởi 24 bit có giá trị từ 0 đến 255 (tương ứng mỗi 8 bit biểu thị cho ba màu cơ bản: đỏ - red, xanh lá – green , xanh dương – blue hoặc tím – magenta, vàng – yellow, và xanh dương - cyan).

Dưới đây, em xin đưa ra mô hình màu RGB. Ta có thể hiểu rằng ảnh RGB sẽ có ba kênh màu là đỏ, xanh lá và xanh dương.

Hình 3. 1 Mô tả ba kênh cơ bản của ảnh màu RGB

Vậy là mỗi pixel của hình ảnh sẽ là sự kết hợp của ba giá trị (đỏ, xanh lá, xanh dương).

Hình 3. 2 Biểu diễn RGB của một pixel

Như ta có thể thấy, với mỗi pixel, ta có ba giá trị có thể biểu diễn bằng số nhị phân. Khi làm việc với các số nhị phân, tương tự với các số thập phân, ta sẽ có những bit có trọng số thấp và trọng số cao. Ví dụ như với giá trị 10110001 (tương ứng giá trị 177 ở số thập phân):

Hình 3. 3 Biểu diễn số thập phân bằng số nhị phân

Các giá trị nằm ở phía trên trái sẽ là những bit có trọng số cao. Nếu ta thay đổi các bit này, giá trị cuối cùng sẽ thay đổi rất lớn. Ví dụ, nếu ta thay đổi bit bên trái ngoài cùng từ **1** sang **0** ( **11111111** sang **01111111**) thì giá trị thập phân của nó sẽ thay đổi từ **255** xuống **177**.

Mặt khác, các bit nằm ở phía bên phải được gọi là những *bit có trọng số thấp*. Nếu ta thay đổi các bit ở phía bên phải này, giá trị thập phân của chúng cũng sẽ có ít sự thay đổi hơn. Ví dụ, nếu ta thay đổi bit bên phải ngoài cùng từ 1 sang 0 (11111111 sang 11111110) thì giá trị thập phân của chúng sẽ thay đổi từ 255 xuống 254. Có nghĩa rằng bit nằm bên phải chỉ thay đổi 1 đơn vị trong khoảng 256 đơn vị (sự thay đổi – sai số nhỏ hơn 1%).

Vậy là, trong một hình ảnh số (ví dụ mô hình màu RGB), mỗi giá trị RGB được biểu diễn bằng 8 bit (8 giá trị thập phân) và các giá trị bit nằm bên phải thì sẽ có trọng số thấp nhất. Vì vậy, nếu ta thay đổi các giá trị nằm phía bên phải này, giá trị màu sắc cũng sẽ thay đổi rất nhỏ, và sẽ tác động rất nhỏ vào màu sắc ở trên hình ảnh gốc. Do những đặc điểm này, ta có thể sử dụng những bit này làm nơi giấu tin (cụ thể là những bit của hình ảnh thủy vân).

* 1. **PHƯƠNG PHÁP THỦY VÂN SỬ DỤNG KĨ THUẬT BIT CÓ TRỌNG SỐ THẤP (LSB)**
     1. ***Mô tả phương pháp Thủy Vân sử dụng kĩ thuật LSB***

### *a,Thuật toán giấu*

*Đầu vào*:

- Ảnh màu ‘I’ với vai trò làm ảnh chứa thủy vân (định dạng PNG).

- Thủy vân ‘W’ (một bức ảnh màu khác với định dạng PNG).

*Đầu ra*:

- Ảnh chứa thủy vân Iw.

*Các bước thực hiện*:

- Bước 1: Kiểm tra xem ảnh chứa thủy vân (I) có kích cỡ bé hơn thủy vân (W) hay không. Nếu ảnh chứa thủy vân có kích cỡ bé hơn thì báo lỗi.

- Bước 2: Hình ảnh I và W được biểu diễn thành các ma trận MxN và PxQ và biến đổi chúng thành mảng 1 chiều có i phần tử.

- Bước 3: Trong pixel giá trị i trong hai mảng I và W, các giá trị sẽ được biểu diễn bằng ma trận [r,g,b] có giá trị nhị phân.

- Bước 4: Cứ 8 bit của từng giá trị trong ma trận [r,g,b], ta sẽ thay thế 4 bit đầu của W vào 4 bit cuối của I.

- Bước 5: Lặp lại từ bước 3 cho đến khi ta đã giấu tin hết giá trị trong từng pixel của W. Nếu vẫn còn dư thừa giá trị pixel của I nhưng đã giấu hết giá trị của W, 4 bit cuối cùng của các pixel còn lại của I sẽ có giá trị ‘0000’. Chuyển đổi ảnh Iw từ mảng một chiều về mảng hai chiều MxN phần tử.

- Bước 6: Trả về hình ảnh Iw là hình ảnh đã giấu thủy vân bên trong.

*Ví dụ minh họa:*

Pixel từ ảnh I:

R(1100 1010)

G(0010 0110)

B(1110 1110)

Pixel từ ảnh W:

R(*0010* 1010)

G(*1100* 0001)

B(*1111* 1110)

Pixel của ảnh mới Iw:

R(1100 *0010*)

G(0010 *1100*)

B(1110 *1111*)

### *b, Thuật toán tách*

*Đầu vào*:

- Ảnh chứa thủy vân Iw.

*Đầu ra*:

- Ảnh chứa thủy vân Iw.

- Thủy vân ‘W’ (một bức ảnh màu với định dạng PNG).

*Các bước thực hiện*:

- Bước 1: Biểu diễn ảnh chứa thủy vân Iw thành ma trận MxN và chuyển đổi chúng thành mảng một chiều có độ dài là i.

- Bước 2: Với mỗi giá trị của ma trận, ta có thể lấy ra một ma trận [r,g,b]. Cứ mỗi giá trị này, ta lấy 4 bit sau, thêm vào chúng là 4 bit có giá trị ‘0000’. Vậy là ta được ma trận [r + ‘0000’, g + ‘0000’, b + ‘0000’].

- Bước 3: Gộp các giá trị này với nhau, ta được một mảng một chiều. Biến đổi mảng đó thành hai ma trận 2 chiều MxN, kiểm tra xem có ma trận nào có nhiều giá trị ‘0000 0000’, ta sẽ crop ma trận đó.

- Bước 4 : Chuyển đổi các giá trị này thành số thập phân. Ta được một hình ảnh W.

*Ví dụ minh họa:*

Pixel của ảnh chứa thủy vân Iw:

R(1100 *0010*)

G(0010 *1100*)

B(1110 *1111*)

Pixel của ảnh W đẩu ra :

R(*0000* 0000)

G(*1100* 0000)

B(*1111* 0000)

* + 1. ***Mô tả phương pháp Thủy Vân sử dụng kĩ thuật LSB và toán tử XOR***

### *a, Thuật toán giấu*

*Đầu vào*:

- Ảnh màu ‘I’ với vai trò làm ảnh chứa thủy vân (định dạng PNG).

- Thủy vân ‘W’ (một bức ảnh màu khác với định dạng PNG).

*Đầu ra*:

- Ảnh chứa thủy vân Iw.

*Các bước thực hiện*:

- Bước 1: Kiểm tra xem ảnh chứa thủy vân (I) có kích cỡ bé hơn thủy vân (W) hay không. Nếu ảnh chứa thủy vân có kích cỡ bé hơn thì báo lỗi.

- Bước 2: Hình ảnh I và W được biểu diễn thành các ma trận MxN và PxQ và biến đổi chúng thành mảng 1 chiều có i phần tử.

- Bước 3: Trong pixel giá trị i trong hai mảng I và W, các giá trị sẽ được biểu diễn bằng ma trận [r,g,b] có giá trị nhị phân.

- Bước 4: Cứ 8 bit của từng giá trị trong ma trận [r,g,b], ta sẽ XOR các bit với thứ tự 1st, 2nd,3rd,4th của cả hai mảng I, W. Giá trị đã được XOR sẽ thay thế 4 bit cuối của I.

- Bước 5: Lặp lại từ bước 3 cho đến khi ta đã giấu tin hết giá trị trong từng pixel của W. Nếu vẫn còn dư thừa giá trị pixel của I nhưng đã giấu hết giá trị của W, 4 bit cuối cùng của các pixel còn lại của I sẽ có giá trị ‘0000’. Chuyển đổi ảnh Iw từ mảng một chiều về mảng hai chiều MxN phần tử.

- Bước 6: Trả về hình ảnh Iw là hình ảnh đã giấu thủy vân bên trong.

*Ví dụ minh họa:*

Pixel từ ảnh I:

R(1100 1010)

G(0010 0110)

B(1110 1110)

Pixel từ ảnh W:

R(*0010* 1010)

G(*1100* 0001)

B(*1111* 1110)

XOR 4 bit đầu của cả hai mảng

1100 XOR *0010 = 1110*

0010 XOR *1100 = 1110*

1110 XOR *1111 = 0001*

Pixel của ảnh mới Iw:

R(1100 *1110*)

G(0010 *1110*)

B(1110 *0001*)

### *b, Thuật toán tách*

*Các bước thực hiện*:

- Bước 1: Biểu diễn ảnh chứa thủy vân Iw thành ma trận MxN và chuyển đổi chúng thành mảng một chiều có độ dài là i.

- Bước 2: Với mỗi giá trị của ma trận, ta có thể lấy ra một ma trận [r,g,b]. Cứ mỗi giá trị này, ta lấy 4 bit sau, XOR với 4 giá trị đầu của 4 bit đầu trong ảnh Iw , thêm vào chúng là 4 bit có giá trị ‘0000’. Vậy là ta được ma trận [r + ‘0000’, g + ‘0000’, b + ‘0000’].

- Bước 3: Gộp các giá trị này với nhau, ta được một mảng một chiều. Biến đổi mảng đó thành hai ma trận 2 chiều MxN, kiểm tra xem có ma trận nào có nhiều giá trị ‘0000 0000’, ta sẽ crop ma trận đó.

- Bước 4 : Chuyển đổi các giá trị này thành số thập phân. Ta được một hình ảnh W.

*Ví dụ minh họa:*

Pixel của ảnh mới Iw:

R(1100 *1110*)

G(0010 *1110*)

B(1110 *0001*)

Pixel của ảnh W đẩu ra :

R(*0010* 0000)

G(*1100* 0000)

B(*1111* 0000)

* 1. **MÔ PHỎNG PHƯƠNG PHÁP THỦY VÂN SỬ DỤNG KĨ THUẬT BIT CÓ TRỌNG SỐ THẤP**

Ngôn ngữ cài đặt, môi trường được thực hiện trên ngôn ngữ lập trình Python. Hệ điều hành được sử dụng là Windows 8.1.

Trong phần mô phỏng này, em sẽ sử dụng tỉ số PSNR để đánh giá chất lượng ảnh gốc với ảnh sau khi đã nhúng thủy vân và ảnh thủy vân trước khi nhúng với ảnh thủy vân sau khi được tách ra khỏi ảnh đã nhúng thủy vân.

* + 1. ***Môi trường cài đặt - Python***

*a, Khái niệm Python:*

- NGÔN NGỮ PYTHON được Guido van Rossum tạo ra cuối năm 1990.

- Python khá giống Perl, Ruby, Scheme, Smalltalk và Tcl.

- Python được phát triển trong một dự án mã mở do một tổ chức phi lợi nhuận Python Software Foundation quản lý.

- Python được phát triển để chạy trên nền Unix. Nhưng theo thời gian, nó đã ‘bành trướng’ sang mọi hệ điều hành từ MS-DOS đến MAC OS, OS/2, Windows, Linux và một số điều hành khác thuộc họ Unix.

- Python là ngôn ngữ bậc cao (high-level) . có hình thức sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình. Cho phép người sử dụng viết mã với số lần gõ phím tối thiểu.

- Python cũng là một trong những ngôn ngữ phổ biến nhất thế giới.

*b, Ưu điểm của Python*

- Cú pháp dễ học, dễ đọc.

- Thư viện phong phú.

- Cộng đồng sử dụng lớn.

*c,Ứng dụng của Python*

- Lập trình web (Youtube, Google, Dropbox, Quora, Reddit, Instagram, NASA, FireFox, Yahoo Maps,…)

- Phân tích dữ liệu

+ Học máy (Machine Learning)

+ Thị giác hóa dữ liệu (Data Visualization)

+ Dữ liệu đám mây (Cloud)

+Phân tích hình ảnh và video (Image and Video Analysis

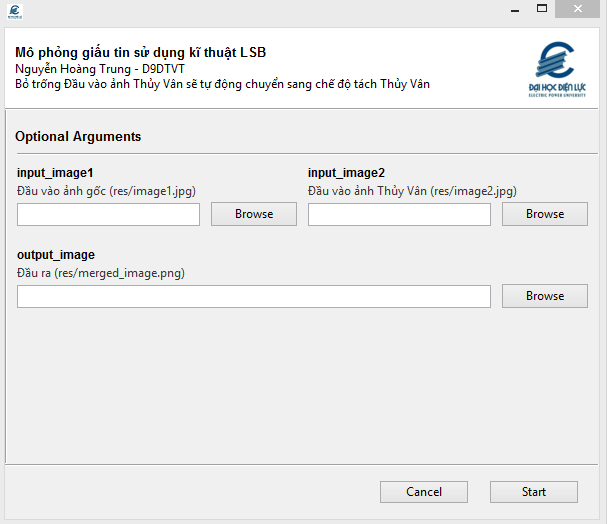
- Lập trình robot

- Lập trình game

- Lập trình ứng dụng

- Bảo mật mạng và máy tính

* + 1. ***Giao diện chương trình chính***

***Dưới đây là một số giao diện của chương trình.***

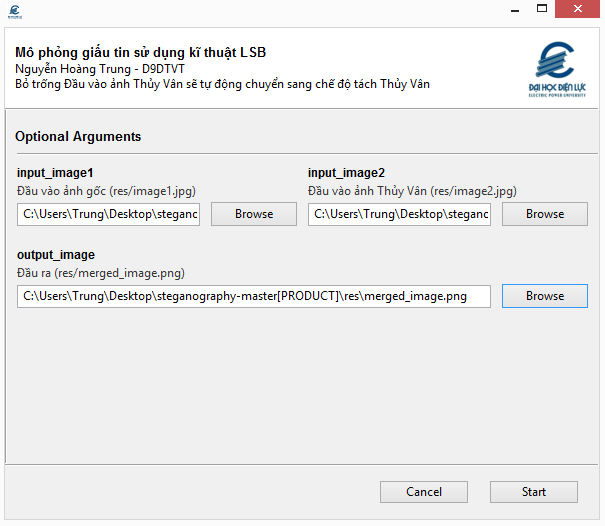
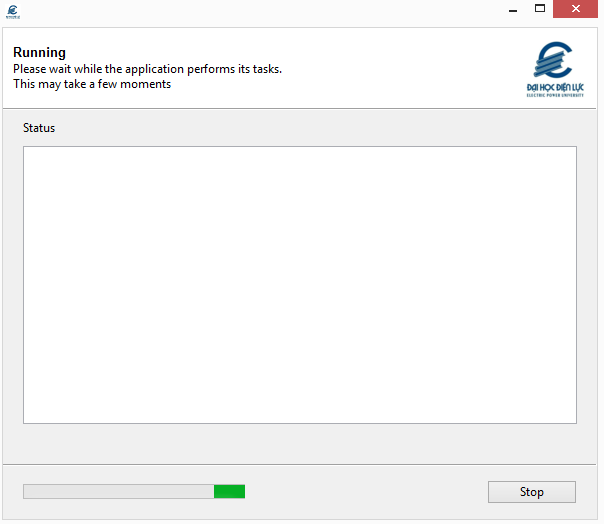
Hình 3. 4 Giao diện chính của chương trình

* Quá trình giấu tin:

- Lựa chọn ảnh gốc từ tệp có sẵn.

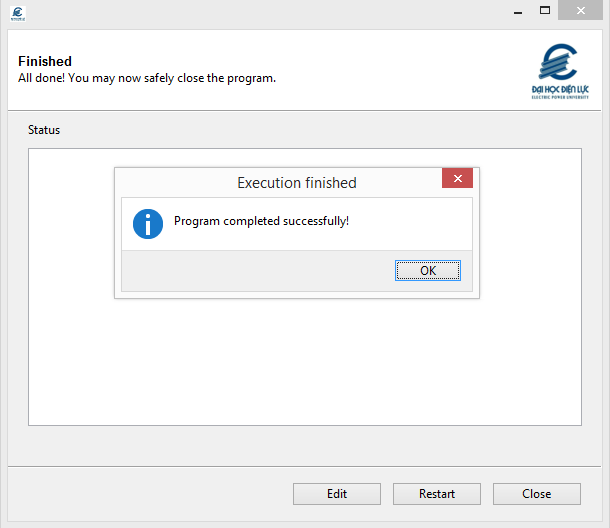
- Lựa chọn Thủy vân, ảnh muốn giấu tin. (Lưu ý là ảnh Thủy vân cần có kích cỡ nhỏ hơn ảnh gốc).

- Lựa chọn đầu ra là nơi ảnh đã giấu tin sẽ được tạo ra, nếu chưa có sẵn, chỉnh sửa tên file ở thư mục cần đặt đầu ra.

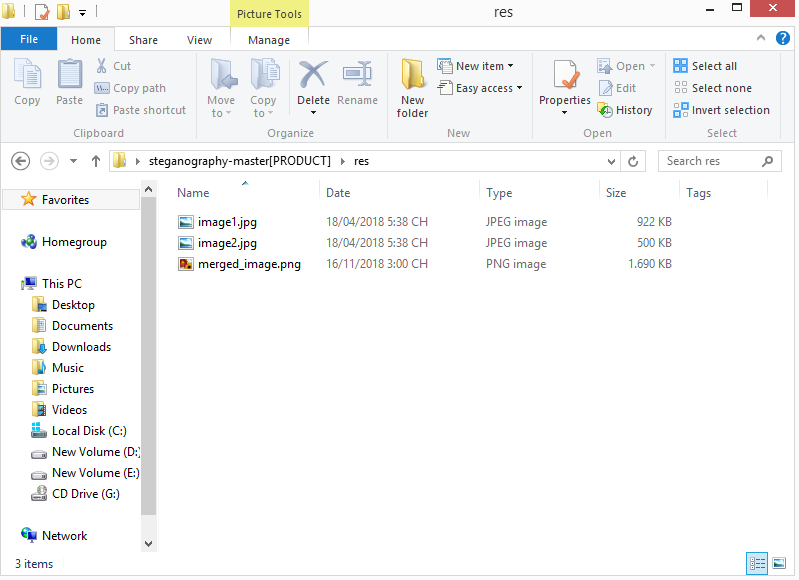
- Sau đó chọn ‘Start’ để chạy chương trình giấu tin.

Hình 3. 6 Giao diện đang chạy của chương trình

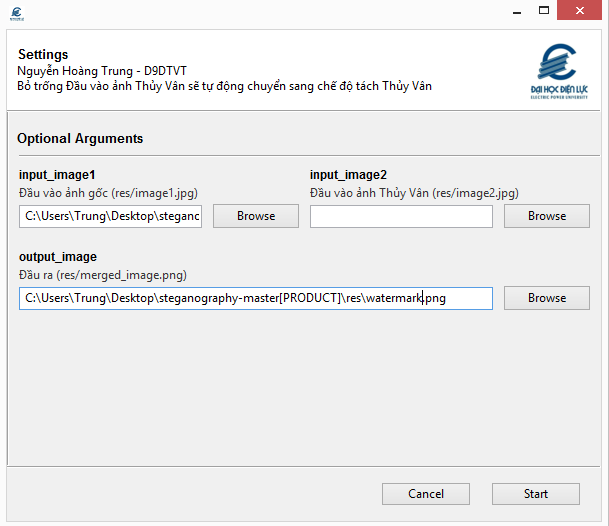
Hình 3. 5 Giao diện trước khi chạy chương trình



Hình 3. 7 Giao diện sau khi hoàn thành chương trình

* Sau khi thực hiện xong chương trình, ta sẽ nhận được thông báo “ Execution finished: Program completed successfully!”. Điều này có nghĩa chương trình đã thực thi thành công.

Hình 3. 8 Hình ảnh được nhúng Thủy Vân

* Ở thư mục đầu ra ta đã chọn, ta sẽ thấy hình ảnh đã được nhúng Thủy vân.

Hình 3. 9 Giao diện ở chế độ tách thủy vân

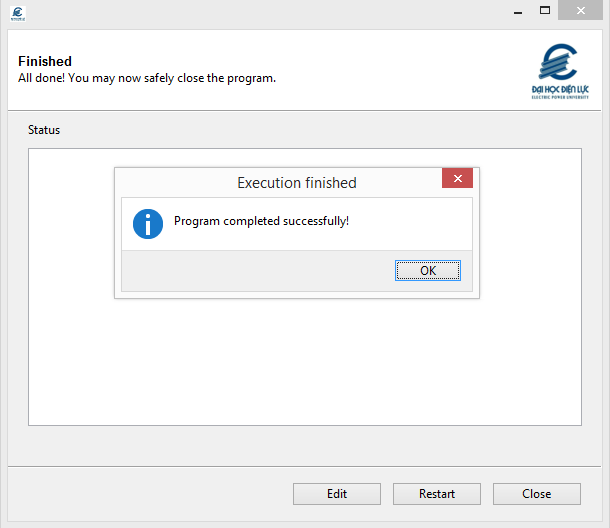
* Quy trình tách Thủy Vân

- Ở đầu vào ảnh gốc, ta lựa chọn ảnh đã giấu Thủy Vân là ảnh gốc.

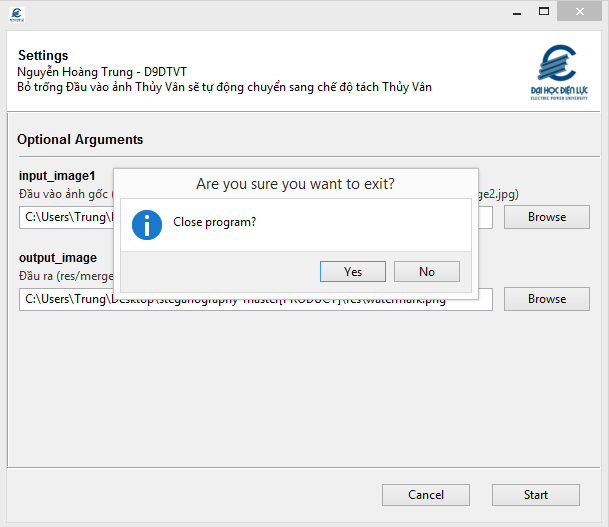
- Ở đầu vào Thủy Vân, ta bỏ trống phần đó, chương trình sẽ tự động nhận biết rằng ta đang ở chế độ tách Thủy Vân.

- Ở đầu ra, ta lựa chọn đầu ra là nơi ảnh đã giấu tin sẽ được tạo ra, nếu chưa có sẵn, chỉnh sửa tên file ở thư mục cần đặt đầu ra.

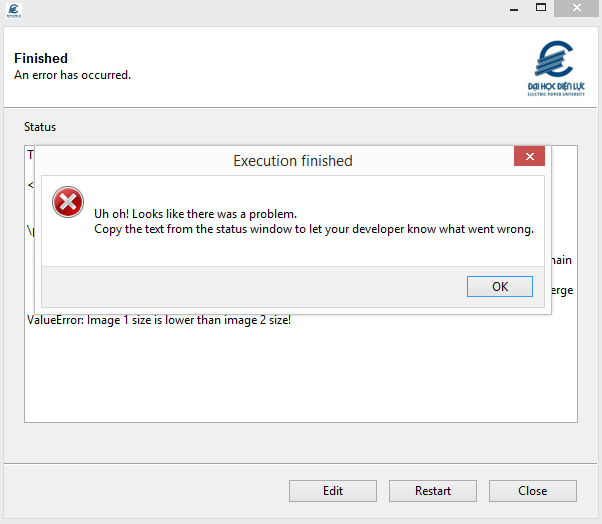
- Sau đó ta nhấn nút ‘Start’ để thực thi chương trình tách Thủy Vân.



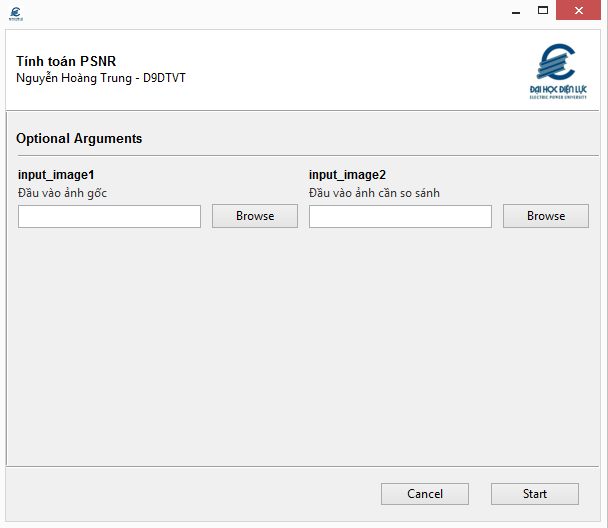
Hình 3. 10 Chương trình tách Thủy vân hoàn thành



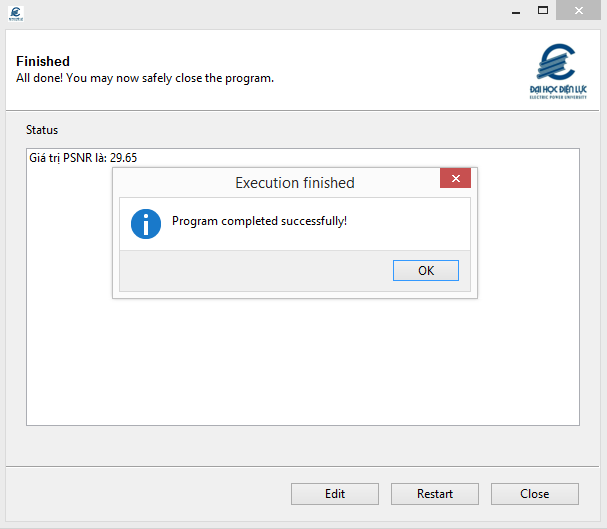
Hình 3. 11 Giao diện khi muốn thoát chương trình



Hình 3. 12 Giao diện chương trình nếu bị lỗi

* + 1. ***Giao diện chương trình PSNR***

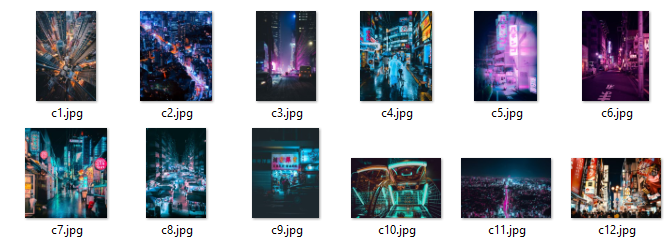
Hình 3. 13 Giao diện chương trình PSNR



Hình 3. 14 Giao diện sau khi chạy xong chương trình

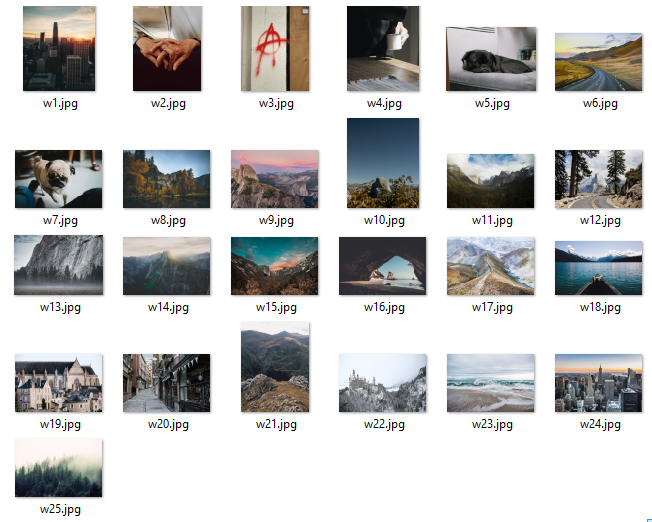
* + 1. ***Thử nghiệm và nhận xét***

*a, Thử nghiệm*

Tập ảnh màu dùng làm tập ảnh gốc bao gồm 12 ảnh JPEG:

Hình 3. 15 Tập ảnh gốc

Tập ảnh màu được dùng làm Thủy Vân:

Dưới đây là bảng kết quả đo PSNR của hai tập ảnh trước và sau khi nhúng Thủy Vân:

Hình 3. 16 Tập ảnh Thủy Vân

Bảng 3. 1 Bảng kết quả đo PSNR trên tập ảnh gốc

| STT | Tên ảnh | Tỉ số PSNR (dB) |
| --- | --- | --- |
| 1 | c1.jpg | 29,14 |
| 2 | c2.jpg | 29,95 |
| 3 | c3.jpg | 29,23 |
| 4 | c4.jpg | 29,4 |
| 5 | c5.jpg | 29,49 |
| 6 | c6.jpg | 30,13 |
| 7 | c7.jpg | 29,82 |
| 8 | c8.jpg | 30,29 |
| 9 | c9jpg | 28,35 |
| 10 | c10.jpg | 29,37 |
| 11 | c11.jpg | 29,23 |
| 12 | c12.jpg | 29,3 |

Bảng 3. 2 Bảng kết quả đo PSNR trên tập ảnh Thủy Vân

| STT | Tên ảnh | Tỉ số PSNR (dB) | STT | Tên ảnh | Tỉ số PSNR (dB) |
| --- | --- | --- | --- | --- | --- |
| 1 | w1.jpg | 29,19 | 14 | w14.jpg | 29.23 |
| 2 | w2.jpg | 28,76 | 15 | w15.jpg | 29,42 |
| 3 | w3.jpg | 29,2 | 16 | w16.jpg | 28,84 |
| 4 | w4.jpg | 29,11 | 17 | w17.jpg | 29,24 |
| 5 | w5.jpg | 29,07 | 18 | w18.jpg | 29,49 |
| 6 | w6.jpg | 30,17 | 19 | w19.jpg | 29,24 |
| 7 | w7.jpg | 29,83 | 20 | w20.jpg | 29,23 |
| 8 | w8.jpg | 29,05 | 21 | w21.jpg | 29,68 |
| 9 | w9jpg | 29,24 | 22 | w22.jpg | 28,6 |
| 10 | w10.jpg | 29,35 | 23 | w23.jpg | 29,16 |
| 11 | w11.jpg | 29,3 | 24 | w24.jpg | 29,21 |
| 12 | w12.jpg | 29,25 | 25 | w25.jpg | 28,97 |
| 13 | w13.jpg | 29,23 |  |  |  |

*b, Nhận xét*

Với kết quả thực nghiệm ở trên, em xin đưa ra một số nhận xét như sau:

- Dễ dàng thực thi: Thuật toán đơn giản, giấu được một lượng lớn thông tin.

- Kĩ thuật được thử nghiệm trên ảnh màu (RGB) với lượng thông tin được giấu tương đối lớn, giá trị độ đo chất lượng ảnh PSNR giữa ảnh gốc và ảnh chứa thủy vân đạt ở mức độ chấp nhận được.

- Mặc dù vậy thì tỉ số PSNR vẫn không phải là một thông số có thể đánh giá sự khác nhau bằng mắt, nó chỉ có thể nói lên một phần sự thay đổi về nhiễu giữa ảnh gốc và ảnh chứa thủy vân. Với tỉ số PSNR càng thấp ta mới có thể nhận ra sự khác nhau giữa hai ảnh (về màu sắc, chất lượng ảnh, chất lượng màu,…). Do vậy, ngoài thông số PSNR, ta cũng cần thực nghiệm bằng mắt người.

- Tuy nhiên Thủy Vân dễ bị can thiệp và ảnh hưởng bởi các tác nhân vì tính đơn giản của thuật toán. Tính bền vững của kĩ thuật chưa thực sự an toàn cho thông tin được giấu. Chính vì vậy, kỹ thuật này nên chỉ áp dụng để xác thực ảnh và làm ví dụ cơ bản thực tế về khái niệm giấu tin.

# KẾT LUẬN

Hiện nay giấu thông tin trong ảnh là một bộ phận chiếm tỉ lệ lớn trong các ứng dụng giấu tin trong đa phương tiện. Lý do là nhu cầu trao đổi thông tin bằng ảnh rất lớn, các kĩ thuật giấu tin trên ảnh cũng đã phát triển và có thành tựu nhất định, giấu tin trong ảnh cũng đóng vai trò hết sức quan trọng trong hầu hết các ứng dụng bảo vệ an toàn thông tin. Chính vì vậy nên vấn đề này nhận được rất nhiều sự quan tâm từ các cá nhân, tổ chức, trường đại học và nhiều viện nghiên cứu trên thế giới.

Trong thời gian làm đồ án, em đã nghiên cứu một số vấn đề liên quan đến thủy vân như sau: Nghiên cứu tổng quan kỹ thuật giấu tin trong ảnh, nghiên cứu các phương pháp thủy vân phổ biến, tìm hiểu về kĩ thuật giấu tin ở những bit ít quan trọng nhất, cài đặt và thử nghiệm chương trình mô phỏng kĩ thuật giấu tin bằng ngôn ngữ Python. Sau khi thử nghiệm, em thấy rằng kĩ thuật và thuật toán không quá phức tạp, cho phép giấu lượng thông tin khá lớn, chất lượng ảnh mặc dù có tổn hao sau khi tách thông tin nhưng đều ở mức chấp nhận được.

Vì thời gian nghiên cứu có hạn, sự hiểu biết của em về vấn đề này sẽ có những hạn chế nhất định nên bài báo cáo của em không tránh khỏi những thiếu sót, em rất mong sự đóng góp ý kiến của các thầy cô giáo để báo cáo của em có thể hoàn thiện hơn nữa

*Hà Nội, ngày 16 tháng 12 năm 2018*

*Sinh viên thực hiện*

*Nguyễn Hoàng Trung*

# TÀI LIỆU THAM KHẢO

[1] Christopher Martin. *Digital Image Watermarking Techniques*, 2000, 1-16.

[2] Upasana Yadav, J.P. Sharma, Dinesh Sharma, Purnima K Sharma. *Different Watermarking Techniques & its Application: A Review*, 2014.

[3] Leung Hon Yin. *Digital Watermarking Device,* 2006.

[4] Rajni Goyal, Naresh Kumar. *LSB Based Digital Watermarking Technique*, 2014.

[5] Nguyễn Diễm Hương. Kĩ thuật giấu tin trên k bit LSB của ảnh, 2012.

[6] Andersen, R.J, Peticolas, F.A.P. *On the limits of steganography*. IEEE Journal of Selected Areas in Communications, Special Issue on Copyright and Privacy Protection 16 No.4, 1998.

[7] <http://ippr-practical.blogspot.com/>

# PHỤ LỤC A. CODE PYTHON CHƯƠNG TRÌNH GIẤU TIN XỬ DỤNG KĨ THUẬT LSB

*“””Sau khi cài đặt Python, ta cần cài đặt thêm một số thư viện: argparse, pillow, gooey”””*

*import argparse*

*from PIL import Image*

*from gooey import Gooey, GooeyParser*

*class Steganography(object):*

*@staticmethod*

*def \_\_int\_to\_bin(rgb):*

*"""*

*Convert an integer tuple to a binary (string) tuple.*

*:param rgb: An integer tuple (e.g. (220, 110, 96))*

*:return: A string tuple (e.g. ("00101010", "11101011", "00010110"))*

*"""*

*r, g, b = rgb*

*return ('{0:08b}'.format(r),*

*'{0:08b}'.format(g),*

*'{0:08b}'.format(b))*

*@staticmethod*

*def \_\_bin\_to\_int(rgb):*

*"""*

*Convert a binary (string) tuple to an integer tuple.*

*:param rgb: A string tuple (e.g. ("00101010", "11101011", "00010110"))*

*:return: Return an int tuple (e.g. (220, 110, 96))*

*"""*

*r, g, b = rgb*

*return (int(r, 2),*

*int(g, 2),*

*int(b, 2))*

*@staticmethod*

*def \_\_merge\_rgb(rgb1, rgb2):*

*"""*

*Merge two RGB tuples.*

*:param rgb1: A string tuple (e.g. ("00101010", "11101011", "00010110"))*

*:param rgb2: Another string tuple (e.g. ("00101010", "11101011", "00010110"))*

*:return: An integer tuple with the two RGB values merged.*

*"""*

*r1, g1, b1 = rgb1*

*r2, g2, b2 = rgb2*

*rgb = (r1[:4] + r2[:4],*

*g1[:4] + g2[:4],*

*b1[:4] + b2[:4])*

*return rgb*

*@staticmethod*

*def merge(img1, img2):*

*"""*

*Merge two images. The second one will be merged into the first one.*

*:param img1: First image*

*:param img2: Second image*

*:return: A new merged image.*

*"""*

*# Check the images dimensions*

*if img2.size[0] > img1.size[0] or img2.size[1] > img1.size[1]:*

*raise ValueError('Image 1 size is lower than image 2 size!')*

*# Get the pixel map of the two images*

*pixel\_map1 = img1.load()*

*pixel\_map2 = img2.load()*

*# Create a new image that will be outputted*

*new\_image = Image.new(img1.mode, img1.size)*

*pixels\_new = new\_image.load()*

*for i in range(img1.size[0]):*

*for j in range(img1.size[1]):*

*rgb1 = Steganography.\_\_int\_to\_bin(pixel\_map1[i, j])*

*# Use a black pixel as default*

*rgb2 = Steganography.\_\_int\_to\_bin((0, 0, 0))*

*# Check if the pixel map position is valid for the second image*

*if i < img2.size[0] and j < img2.size[1]:*

*rgb2 = Steganography.\_\_int\_to\_bin(pixel\_map2[i, j])*

*# Merge the two pixels and convert it to a integer tuple*

*rgb = Steganography.\_\_merge\_rgb(rgb1, rgb2)*

*pixels\_new[i, j] = Steganography.\_\_bin\_to\_int(rgb)*

*return new\_image*

*@staticmethod*

*def unmerge(img):*

*"""*

*Unmerge an image.*

*:param img: The input image.*

*:return: The unmerged/extracted image.*

*"""*

*# Load the pixel map*

*pixel\_map = img.load()*

*# Create the new image and load the pixel map*

*new\_image = Image.new(img.mode, img.size)*

*pixels\_new = new\_image.load()*

*# Tuple used to store the image original size*

*original\_size = img.size*

*for i in range(img.size[0]):*

*for j in range(img.size[1]):*

*# Get the RGB (as a string tuple) from the current pixel*

*r, g, b = Steganography.\_\_int\_to\_bin(pixel\_map[i, j])*

*# Extract the last 4 bits (corresponding to the hidden image)*

*# Concatenate 4 zero bits because we are working with 8 bit values*

*rgb = (r[4:] + "0000",*

*g[4:] + "0000",*

*b[4:] + "0000")*

*# Convert it to an integer tuple*

*pixels\_new[i, j] = Steganography.\_\_bin\_to\_int(rgb)*

*# If this is a 'valid' position, store it*

*# as the last valid position*

*if pixels\_new[i, j] != (0, 0, 0):*

*original\_size = (i + 1, j + 1)*

*# Crop the image based on the 'valid' pixels*

*new\_image = new\_image.crop((0, 0, original\_size[0], original\_size[1]))*

*return new\_image*

*@Gooey(*

*show\_config=True, # skip config screens all together*

*default\_size=(610, 530), # starting size of the GUI*

*program\_name='Mô phỏng giấu tin sử dụng kĩ thuật LSB', # Defaults to script name*

*image\_dir='C:/Users/Trung/Desktop/steganography-master[PRODUCT]/icon') #Icon for the program*

*def main():*

*# Construct the argument parse and parse the arguments*

*#ap = argparse.ArgumentParser()*

*ap = GooeyParser(description="Nguyễn Hoàng Trung - D9DTVT \nBỏ trống Đầu vào ảnh Thủy Vân sẽ tự động chuyển sang chế độ tách Thủy Vân")*

*ap.add\_argument("--input\_image1", type=str, required=True, help="Đầu vào ảnh gốc (res/image1.jpg)", widget='FileChooser')*

*ap.add\_argument("--input\_image2", type=str, required=False, help="Đầu vào ảnh Thủy Vân (res/image2.jpg)", widget='FileChooser')*

*ap.add\_argument("--output\_image", type=str, required=True, help="Đầu ra (res/merged\_image.png)", widget='FileChooser')*

*args = ap.parse\_args()*

*# If the input\_image2 argument is valid (not empty), the user*

*# is trying to merge two images, so call the merge method*

*if args.input\_image2:*

*img1 = Image.open(args.input\_image1)*

*img2 = Image.open(args.input\_image2)*

*merged\_image = Steganography.merge(img1, img2)*

*merged\_image.save(args.output\_image)*

*# Else, try to unmerge the image*

*else:*

*img = Image.open(args.input\_image1)*

*unmerged\_image = Steganography.unmerge(img)*

*unmerged\_image.save(args.output\_image)*

*if \_\_name\_\_ == "\_\_main\_\_":*

*main()*

# PHỤ LỤC B. CODE PYTHON CHƯƠNG TRÌNH TÍNH TOÁN PSNR

*“”” Trước khi chạy chương trình, ta cần cài đặt một số thư viện: argparse, gooey, opencv-python, numpy”””*

*import argparse*

*from gooey import Gooey, GooeyParser*

*import cv2 as cv*

*import numpy as np*

*import math*

*def psnr(img1,img2):*

*image1=cv.imread(img1)*

*image2=cv.imread(img2)*

*mse = np.mean((np.subtract(image2, image1))\*\*2)*

*sqrtmse=math.sqrt(mse)*

*#psnr*

*psnr = 20\*math.log10(255/sqrtmse)*

*print('Giá trị PSNR là: %s' %(round(psnr,2)))*

*return psnr*

*@Gooey(*

*show\_config=True, # skip config screens all together*

*default\_size=(610, 530), # starting size of the GUI*

*program\_name='Tính toán PSNR', # Defaults to script name*

*image\_dir='C:/Users/Trung/Desktop/steganography-master[PRODUCT]/icon') #Icon for the program*

*def main():*

*# Construct the argument parse and parse the arguments*

*#ap = argparse.ArgumentParser()*

*ap = GooeyParser(description="Nguyễn Hoàng Trung - D9DTVT")*

*ap.add\_argument("--input\_image1", type=str, required=True, help="Đầu vào ảnh gốc", widget='FileChooser')*

*ap.add\_argument("--input\_image2", type=str, required=False, help="Đầu vào ảnh cần so sánh", widget='FileChooser')*

*args = ap.parse\_args()*

*psnr(args.input\_image1,args.input\_image2)*

*if \_\_name\_\_ == "\_\_main\_\_":*

*main()*