ĐẠI HỌC QUỐC GIA TP.HCM

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

******

**Môn học**

**ẨN THÔNG TIN TRÊN DỮ LIỆU SỐ VÀ ỨNG DỤNG**

**Đề tài:**

**ĐÁNH DẤU DỮ LIỆU TRÊN HỆ THỐNG THÔNG TIN ĐẠI LÝ:**

**ẨN THÔNG TIN TRONG BẢN ĐỒ VECTOR 2D**

**Giảng viên:** TS. DƯƠNG MINH ĐỨC

**Học viên thực hiện:**

TRẦN THÁI BÌNH - CH1702015

NGUYỄN QUANG ĐẠT - CH1702026

LÊ GIAI TỰ - CH1702048

***TP. Hồ Chí Minh, tháng 08 năm 2019***

ĐẠI HỌC QUỐC GIA TP.HCM

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

******

**Môn học**

**ẨN THÔNG TIN TRÊN DỮ LIỆU SỐ VÀ ỨNG DỤNG**

**Đề tài:**

**ĐÁNH DẤU DỮ LIỆU TRÊN HỆ THỐNG THÔNG TIN ĐẠI LÝ:**

**ẨN THÔNG TIN TRONG BẢN ĐỒ VECTOR 2D**

**Giảng viên:** TS. DƯƠNG MINH ĐỨC

**Học viên thực hiện:**

TRẦN THÁI BÌNH - CH1702015

NGUYỄN QUANG ĐẠT - CH1702026

LÊ GIAI TỰ - CH1702048

***TP. Hồ Chí Minh, tháng 06 năm 2019***

# Giới thiệu

Ngày nay cùng với sự phát triển của internet và truyền thông thông tin, việc bảo vệ bản quyền, truyền thông điệp mật, chống sao chép … trở nên ngày càng quang trọng.

Mặt dù có rất nhiều cách nhúng vô thông điệp vô hình ảnh như miền không gian, miền Cosin (DCT), Wavelet (DWT), … Tất cả các kỹ thuật đều có một yêu cầu chung là tăng cường bảo mật để tránh phát hiện, người sử dụng sẽ không nhìn thấy thông điệp nhúng, có thể nén được nhiều thông tin và đảm bảo trước các cuộc tấn công như nén, xoay, …

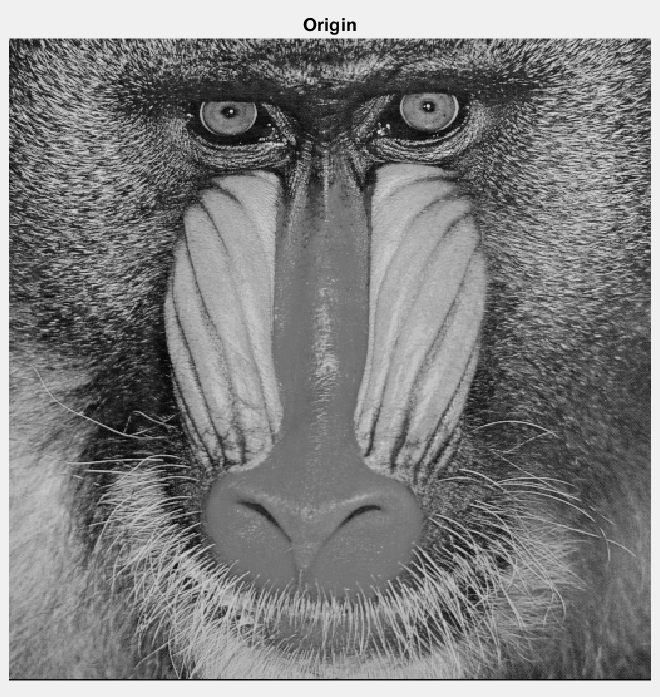
Kỹ thuật ẩn thông tin chia làm hai hướng: ẩn trực tiếp (trên miền không gian – thời gian), và ẩn trên miền biến đổi. Có nhiều miền đặc trưng để lựa chọn làm miền ẩn dữ liệu như biến đổi Cosin (DCT), Wavelet (DWT), ... Bên cạnh đó việc trích xuất cũng được chia làm hai hướng là mù và bán mù.

Việc ân thông tin là một trong bài toán vô cùng phong phú và thú vị. Vì vậy việc tìm hiểu và thực hành tấn công sẽ giúp chúng ta hiểu rõ hơn.

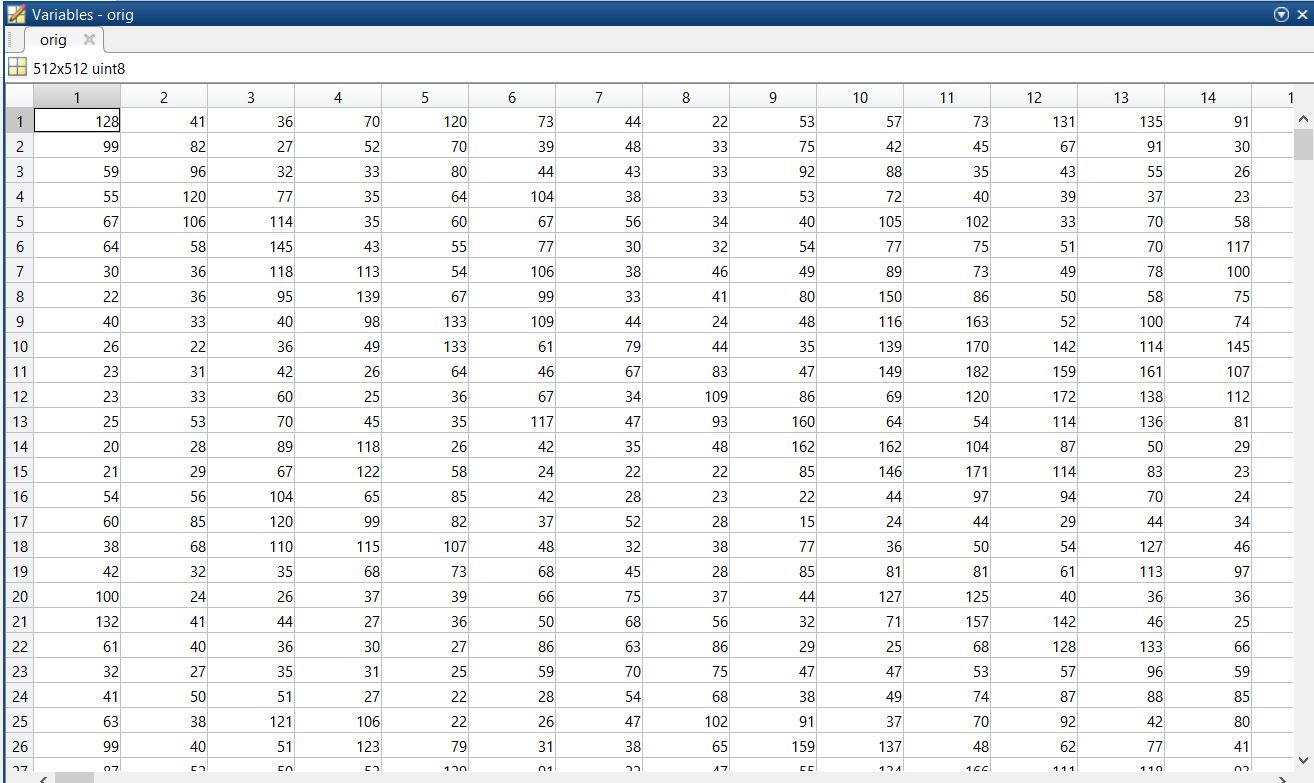
# Ẩn thông tin trong miền không gian (Least Significant Bit)

## Môt tả thuật toán:

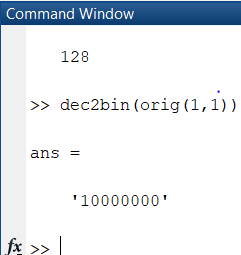
Trong mỗi bức ảnh xám các điểm được thể hiện bằng các trọng số 0 tới 255 (hình 1-a). Từ các trọng số đó sẽ biến đổi thể hiện dưới dạng 8bit (hình 2), dựa vào các bit ta có thể thấy hình ảnh có nét rõ dần từ trọng số thấp tới trọng số cao như hình 1 bên dưới (hình 3).



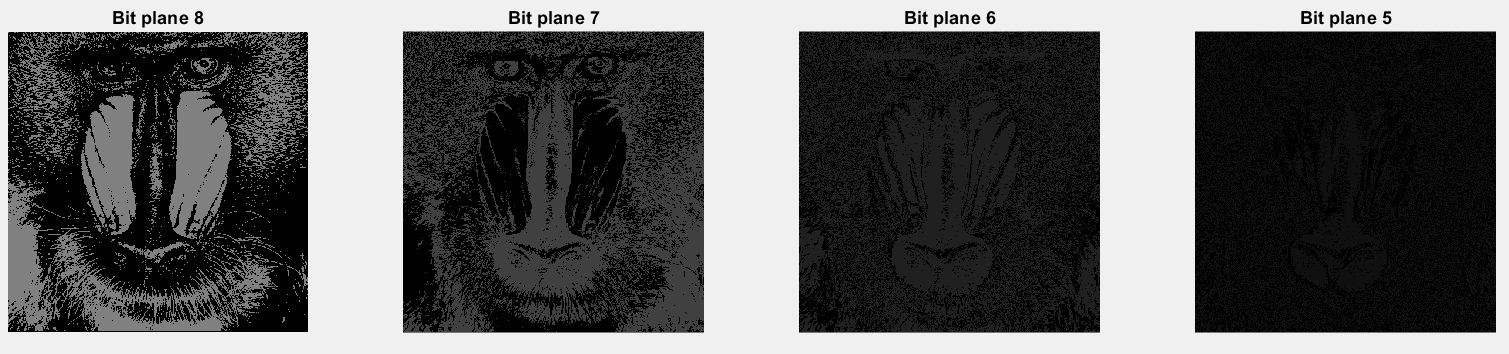
Hình 1 – a. Hình vật mang.

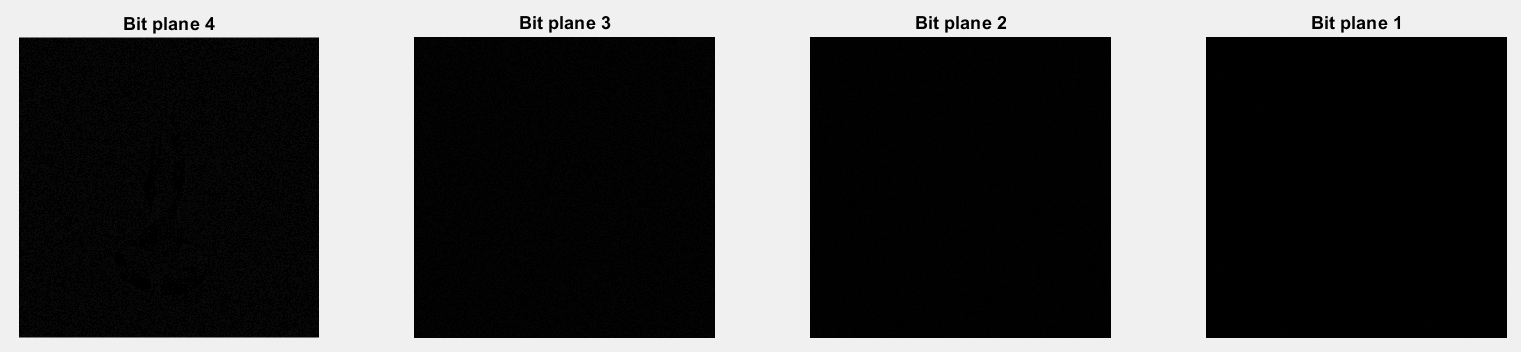


Hình 1 – b. Thể hiện các trọng số của ảnh vật mang



Hình 2. Thển hiện trọng số ảnh dưới dạng bit





Hình 3 – Hình ảnh vật mang dưới bit ở vị trí 8 tới vị trí 1.

Ở hình bit plane 8 là các điểm mà có bit ở vị trí thứ 8 là 1 (2^7 = 128). Do trích xuất ở vị trí trọng số cao nên hình ảnh trở nên rõ rang.

Ở hình bit plane 3 là các điểm mà có bit ở vị trí thứ 3 là 1 (2^2 = 4). Do trích xuất ở vị trí trọng số thấp nên hình ảnh không thấy được mặt con khỉ.

Từ những nhận xét trên ta có được điều như sau: ảnh mà có vị trí bit cao thì sẽ nhìn rõ vật hơn vị trị bit thấp, đồng thời có nghĩa là nếu muốn giấu thông tin thì phải nhét các dữ liệu vào những bit có trọng số thấp.

Từ đó, ta có được thuật toán LSB như sau:

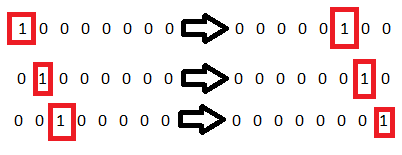
Bước 1: Đọc ảnh gốc và ảnh nhúng.

Bước 2: Lấy các ảnh từ bit 1 tới bit 8 của gốc. Đặt là B1, B2, B3, B4, B5, B6, B7, B8.

Bước 3: Lấy các ảnh từ bit 1 tới bit 8 của nhúng. Đặt là BW1, BW2, BW3, BW4, BW5, BW6, BW7, BW8.

Bước 4: Watermarking = B8+B7+B6+B5+B4+BW8/2^5+BW7/2^5+BW6/2^5;

Ta thấy “BW8/2^5”, “BW7/2^5”, “BW6/2^5” là các điểm mà có chưa bit 8 của thông điệp chuyển thành bit 3, điểm bit 7 chuyển thành bit 2, điểm bít 6 chuyển thành bit 1 (hình 5). Vì các bit nay ở trọng số cao nên khi xuất hình ảnh sẽ nhìn rõ hơn (hình 6).



Hình 5. Mình họa việc chuyển đổi bit



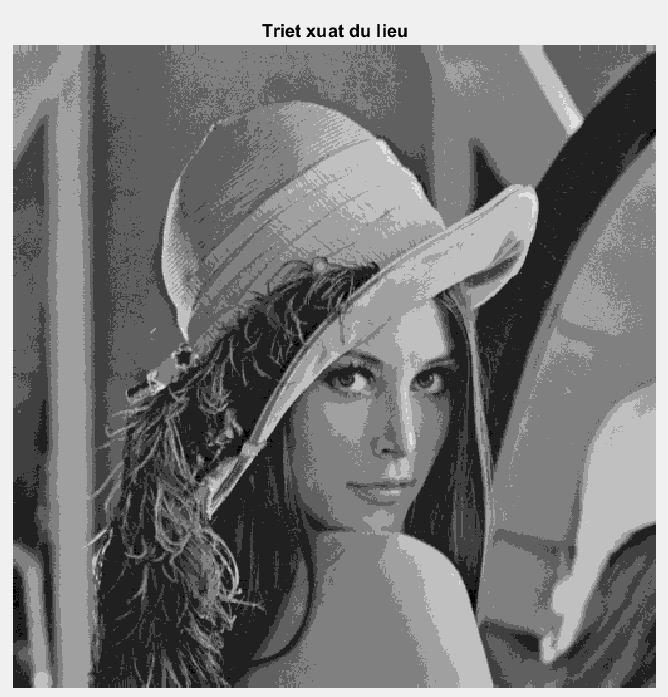
Hình 6. Hình ảnh gộp bit tại 8, 7,6 của thông điệp



Hình 7. Ảnh Watermarked

Bước 6: Thực hình trích xuất dữ liệu bằng cách lấy các ảnh từ bit 1 tới bit 3.

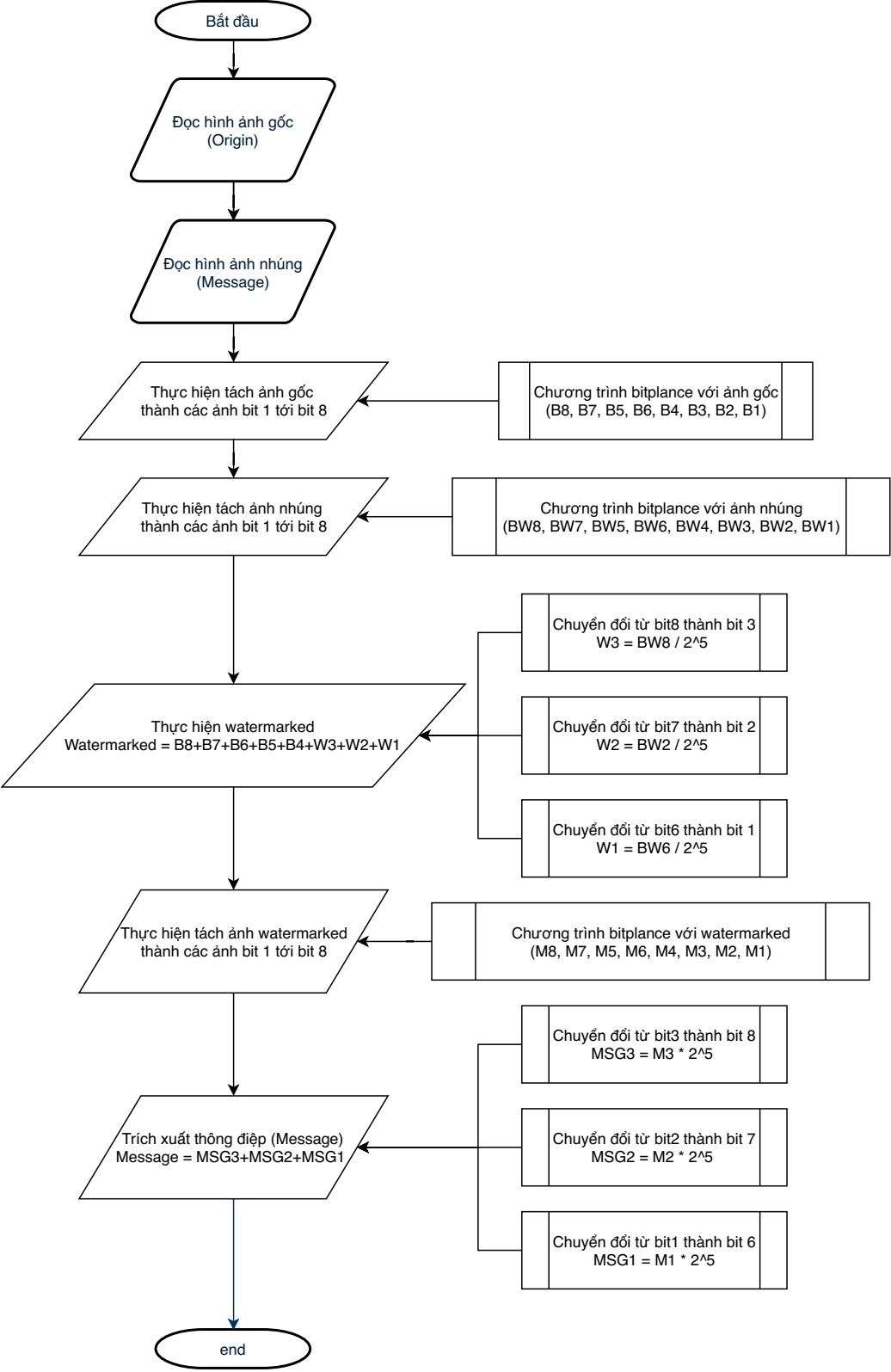
message = W3 \* 2^5 + W2 \* 2^5 + W1 \* 2^5;



Hình 8. Ảnh thông điệp triết xuất

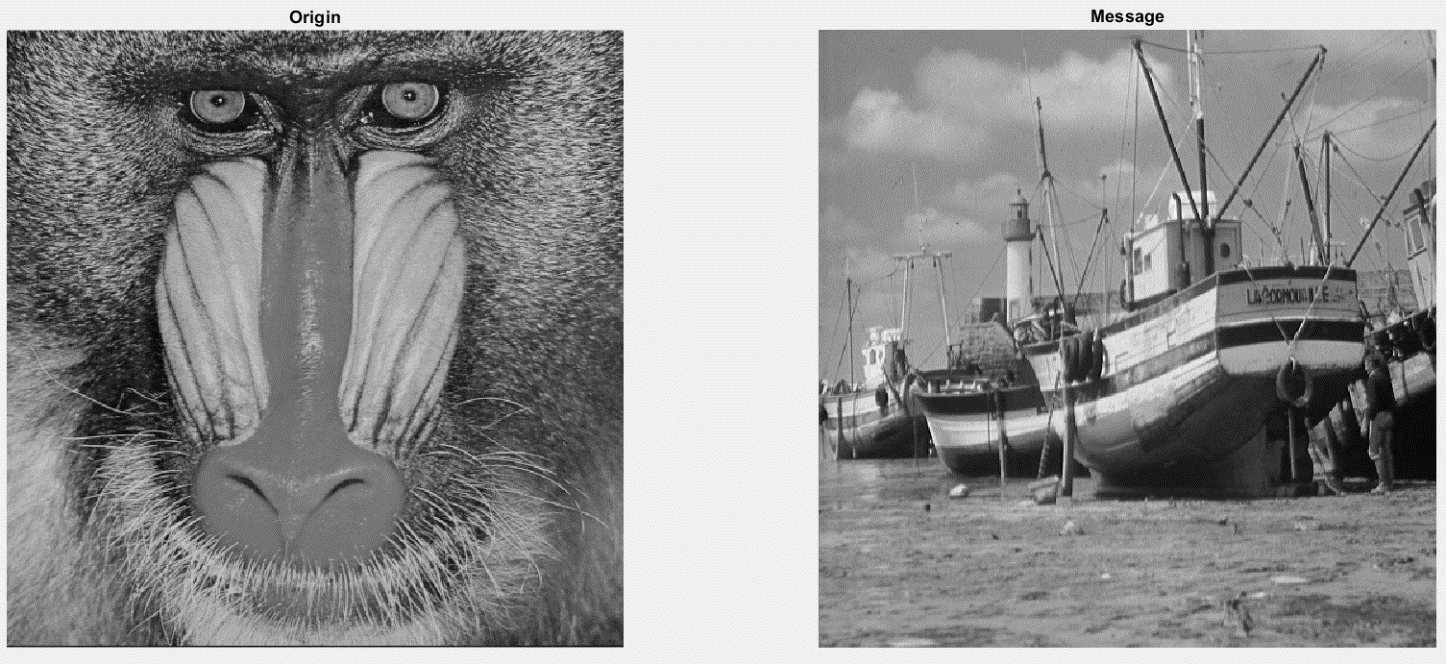
***Chú ý***: Bài toán trên đang sử dụng hai bức ảnh có kích thước cùng nhau. Nếu mốn sử dụng bước ảnh có kích thước khác nhau thì điều kiện là kích thước vật mang phải nhỏ hơn. Sau đó chuyển các điểm có bit 3, 2,1 của vật mang thành 0. Chạy vòng lặp với hình thông điệp để tìm những điểm có bit 8 thì nhét vào bit 3 của vật mang, bit 7 thành bit 2 của vật mang, bit 6 thành bit 1 của vật mang.

## Sơ đồ quá trình nhúng:

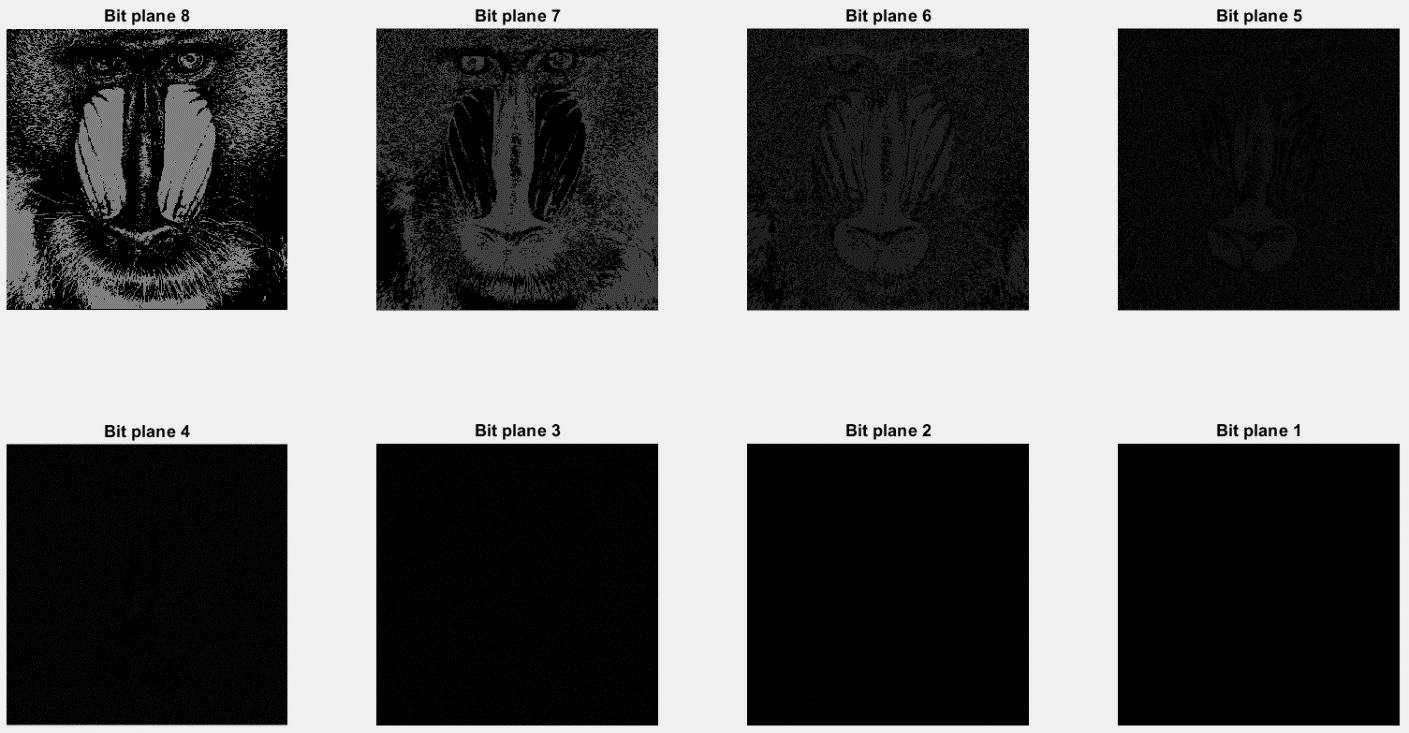


## Kết quả thực nghiệm

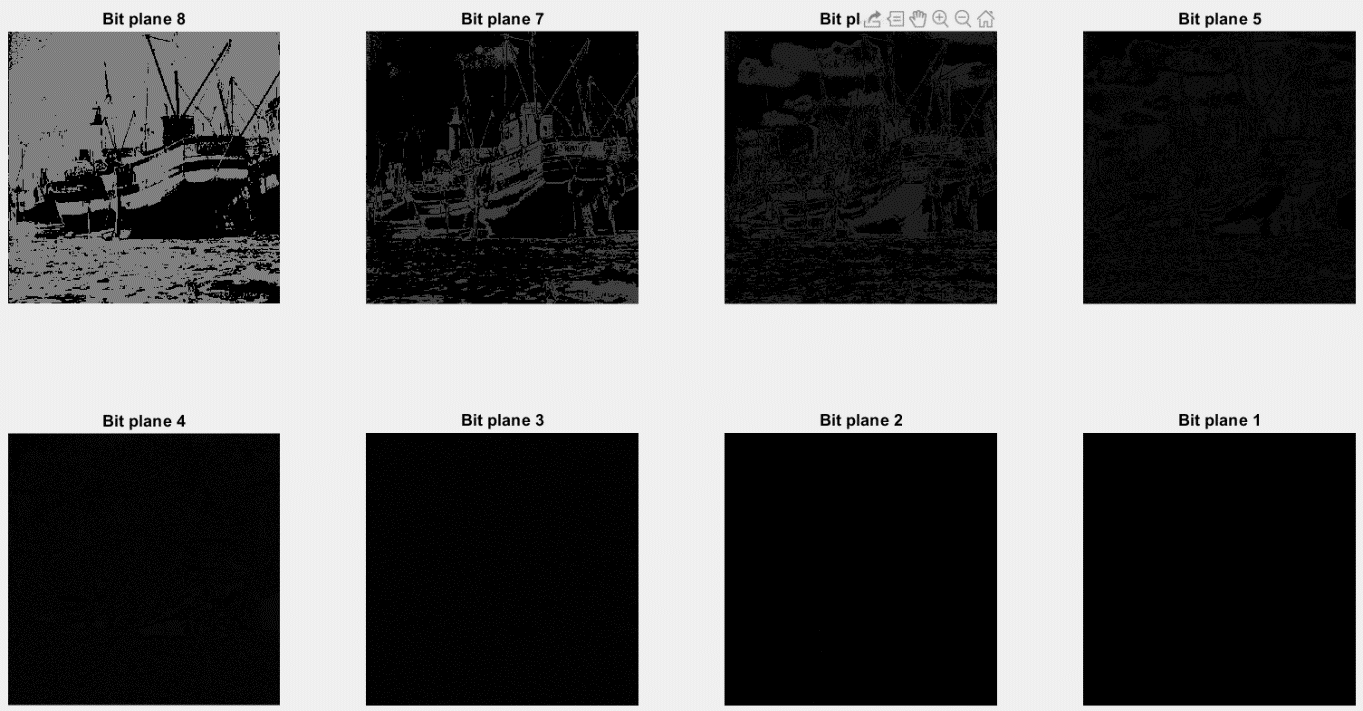
* Đọc hình ảnh gốc và thông điệp nhúng



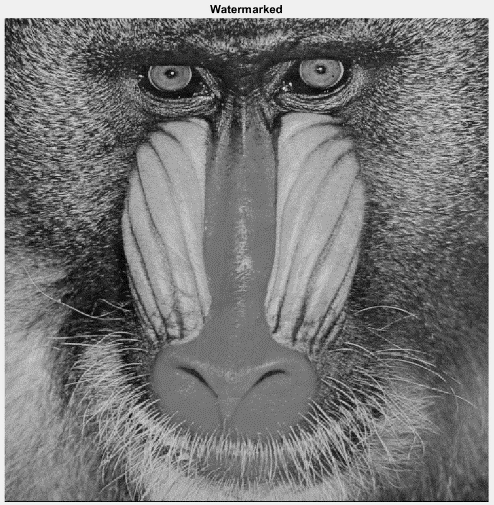
* Tách hình ảnh gốc dưới dạng bit 8, bit 7, bit 6, bit 5, bit 4, bit 3, bit 2, bit 1.



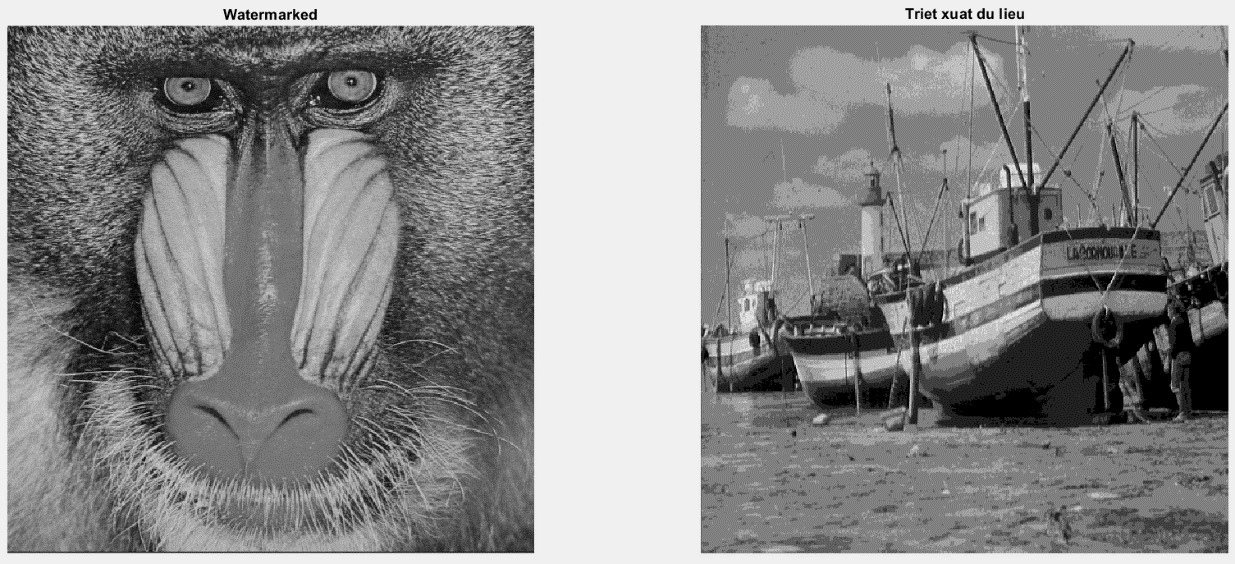
* Tách hình ảnh message dưới dạng bit 8, bit 7, bit 6, bit 5, bit 4, bit 3, bit 2, bit 1.



* Kết quả nhúng LSB



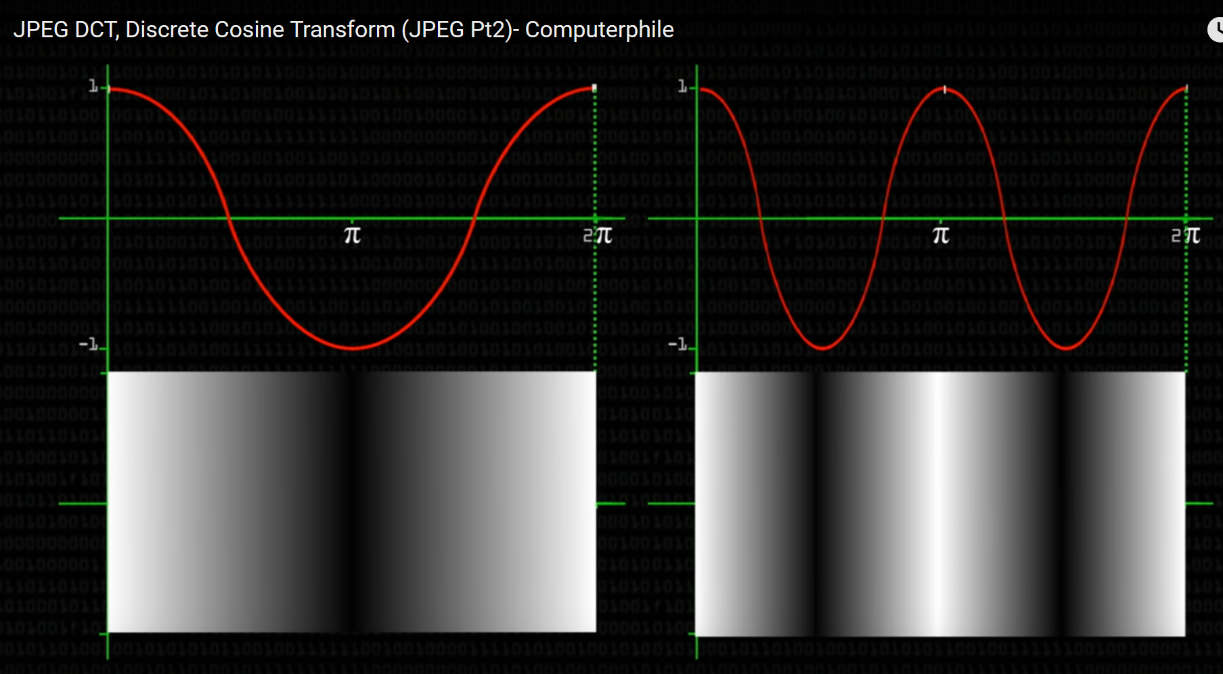
* Message trích xuất

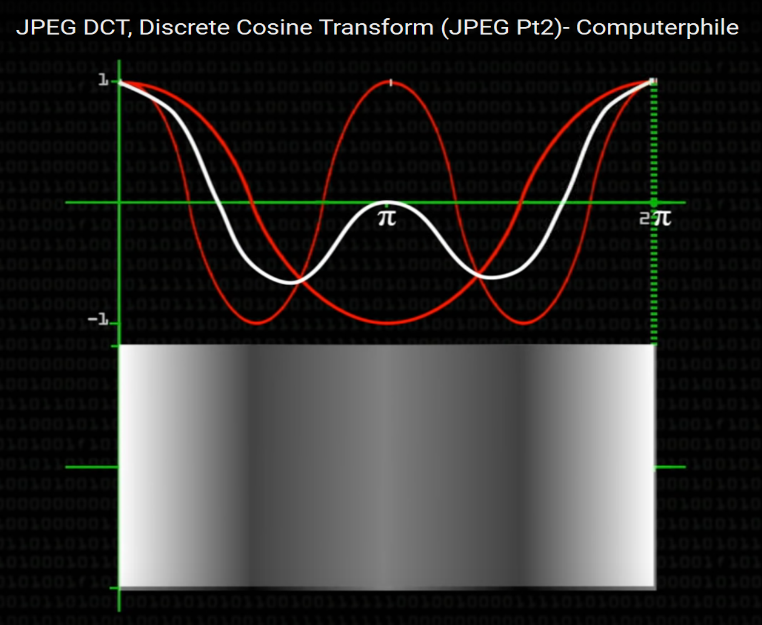


# Ẩn thông tin trong miền phổ (Discrete Cosines Transform)

## Mô tả thuật toán:

Đây là thuật toán ẩn thông tin trong miền cosin, thuật toán này sẽ ảnh hưởng tới độ chói của ảnh sau khi được nhúng (hình 9). Ảnh sau khi nhúng sẽ được trích xuất dưới dạng bán mù, nghĩa là sẽ cần có key để tìm ra điểm trích xuất. Đây là một điểm khách của nó so với thuật toán LSB (trích xuất mù).



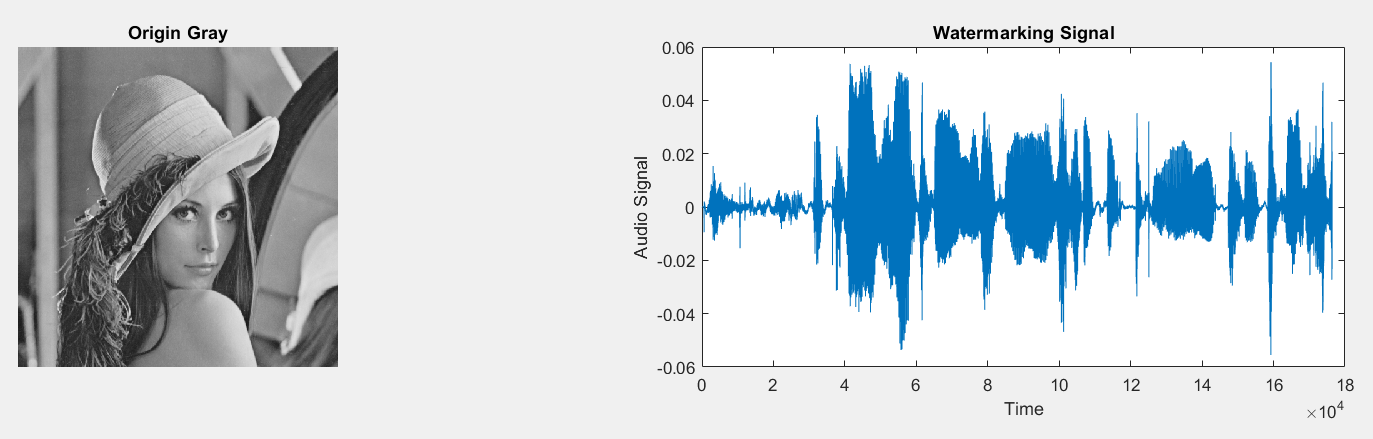


Hình 9. Ví dụ nhúng DCT

Thuật toán này thích hợp cho việc nhúng các tin hiệu âm thanh vào trong audio hoặc hình ảnh. Đây là một thuật toán được phát minh vào năm 1992 bởi Andrew Tirkel và Charles Osborne.

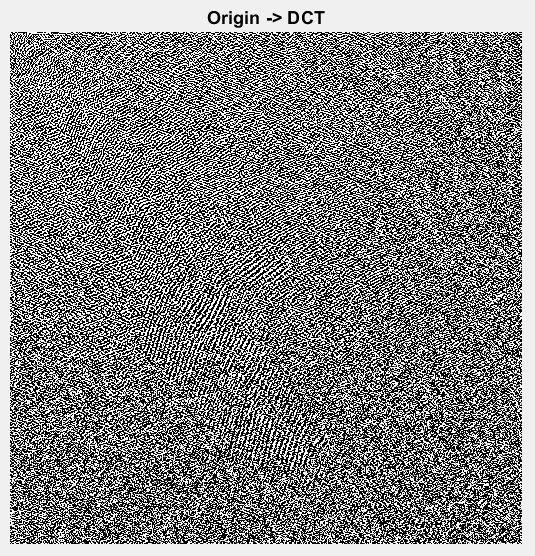
Thuật toán Discrete Cosines Transform hoạt động như sau:

Bước 1: Đọc hình ảnh gốc (kích thước mxn) và âm thanh Message (kích thước 1xn).



Hình 10. Đọc hình ảnh vật mang và signal nhúng.

Bước 2: Chuyển đổi ảnh gốc về miền DCT



Hình 11. Chuyển đổi origin sang DCT

Bước 3: Chuyển đổi ma trận ảnh 2D từ “m x n” -> “1 x (m\*n)”. Sau đó sắp xếp theo thứ tự giảm dần.

Bước 4: Lấy giá trị trong mảng trên bằng với kích thước độ dài âm thanh.

Bước 5: Tìm vị trí của các phẩn tử đã lấy trong vật mang.

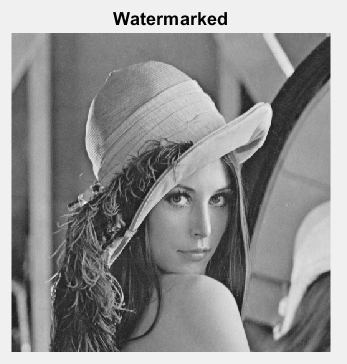
Bước 6: Thực hiện hiện nhúng tại các vị trí đã tìm.

DCT của Watermarked = DCT của Origin \* (1 + a \* Message Signal)

Trong đó a = 25.0, là tỉ lệ nén trọng số thường là từ 10 tơi 50.

Bước 7: Chuyển đổi từ miền DCT sang image ta được Watermarked (hình 12)

Watermarked = dct2(DCT của Watermarked)

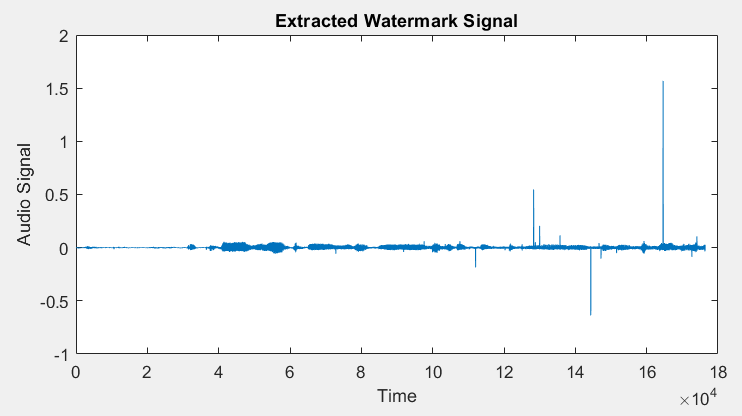


Hình 12. Ảnh đã nhúng thông điệp

Bước 8: Chuyển ảnh Watermarked về DCT

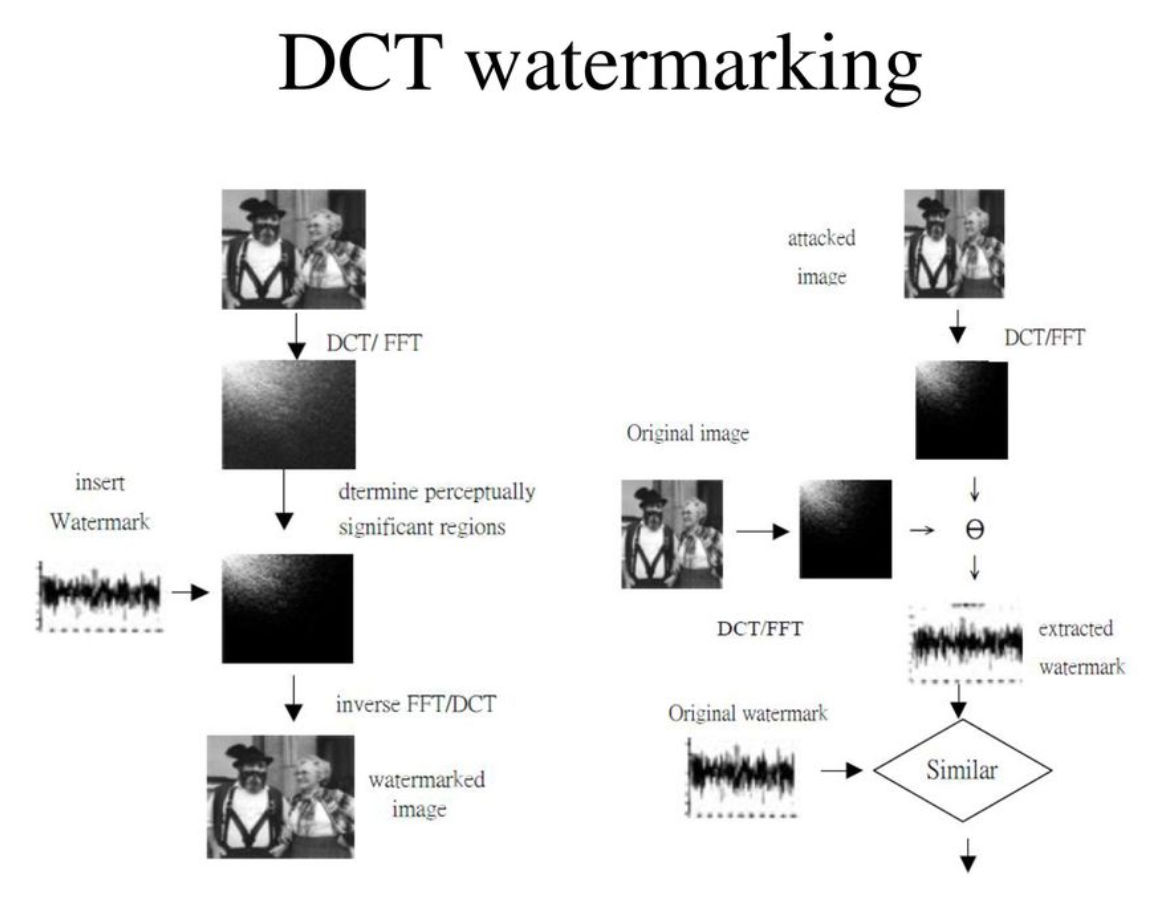
Bước 9: Tiến hành trích xuất message signal dựa vào các điểm đánh dấu nhúng.

M = (DCT của Watermarked / DCT của Origin – 1) / 25.0



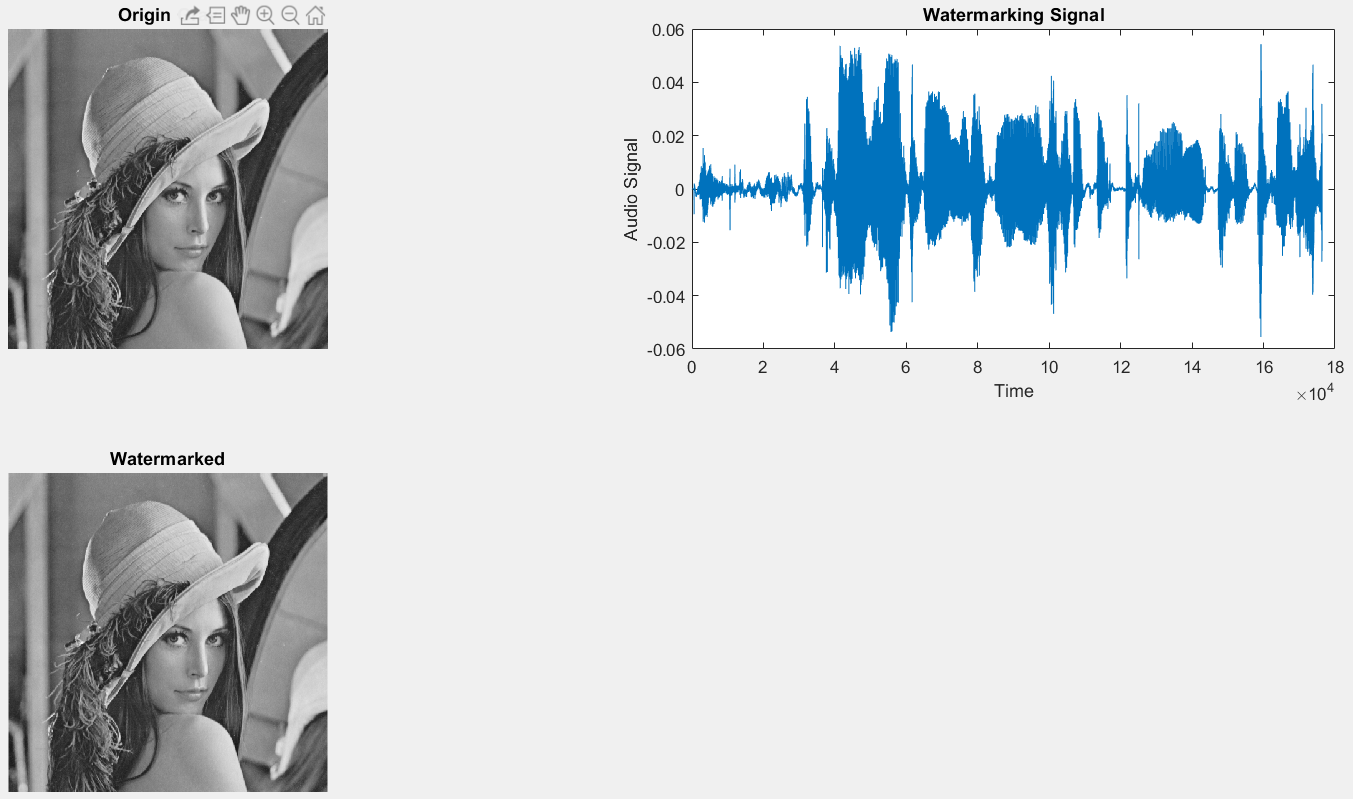
Hình 13. Message trích xuất từ Watermarked

## Sơ đồ quá trình nhúng:

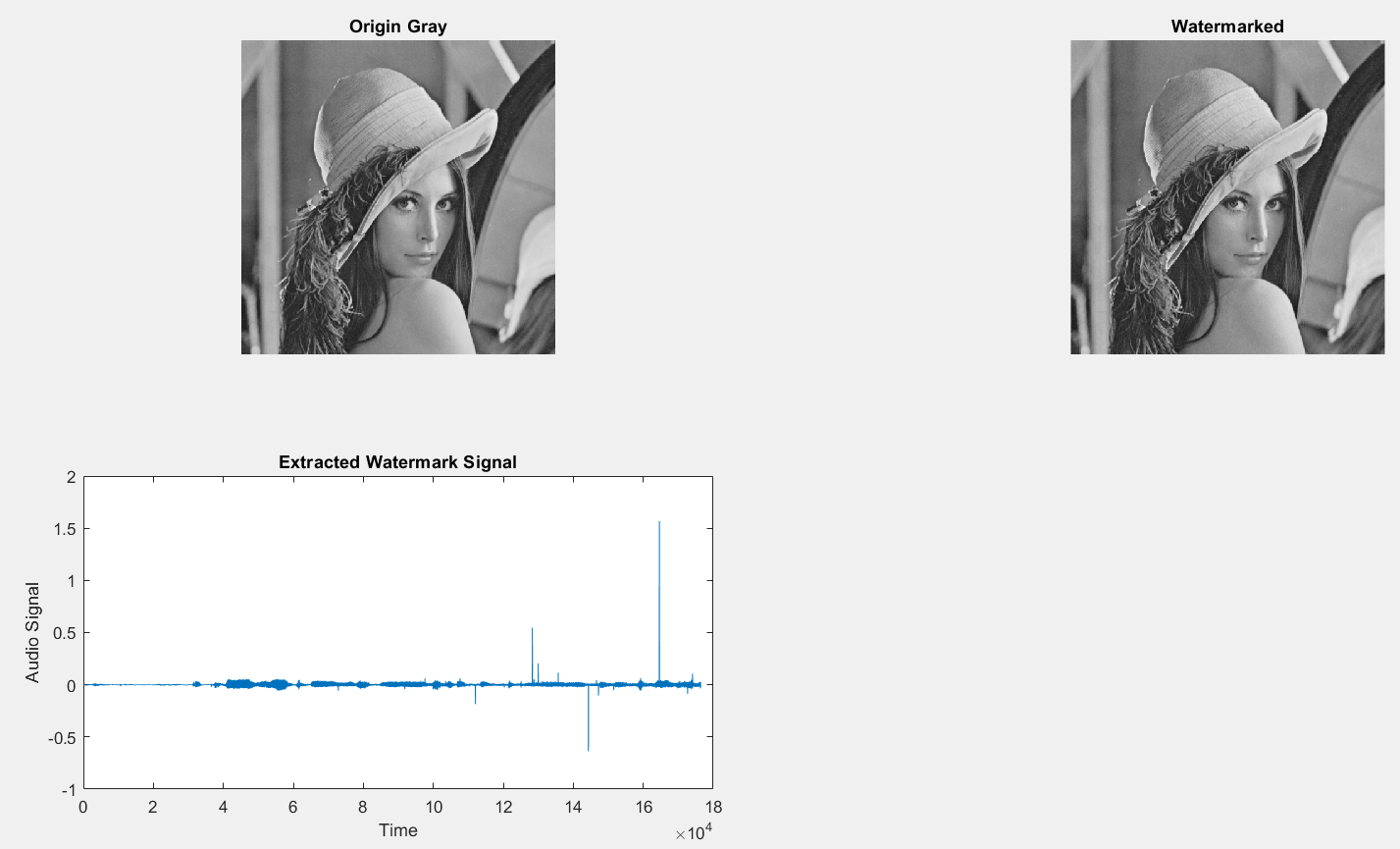


## Kết quả thực nghiệm

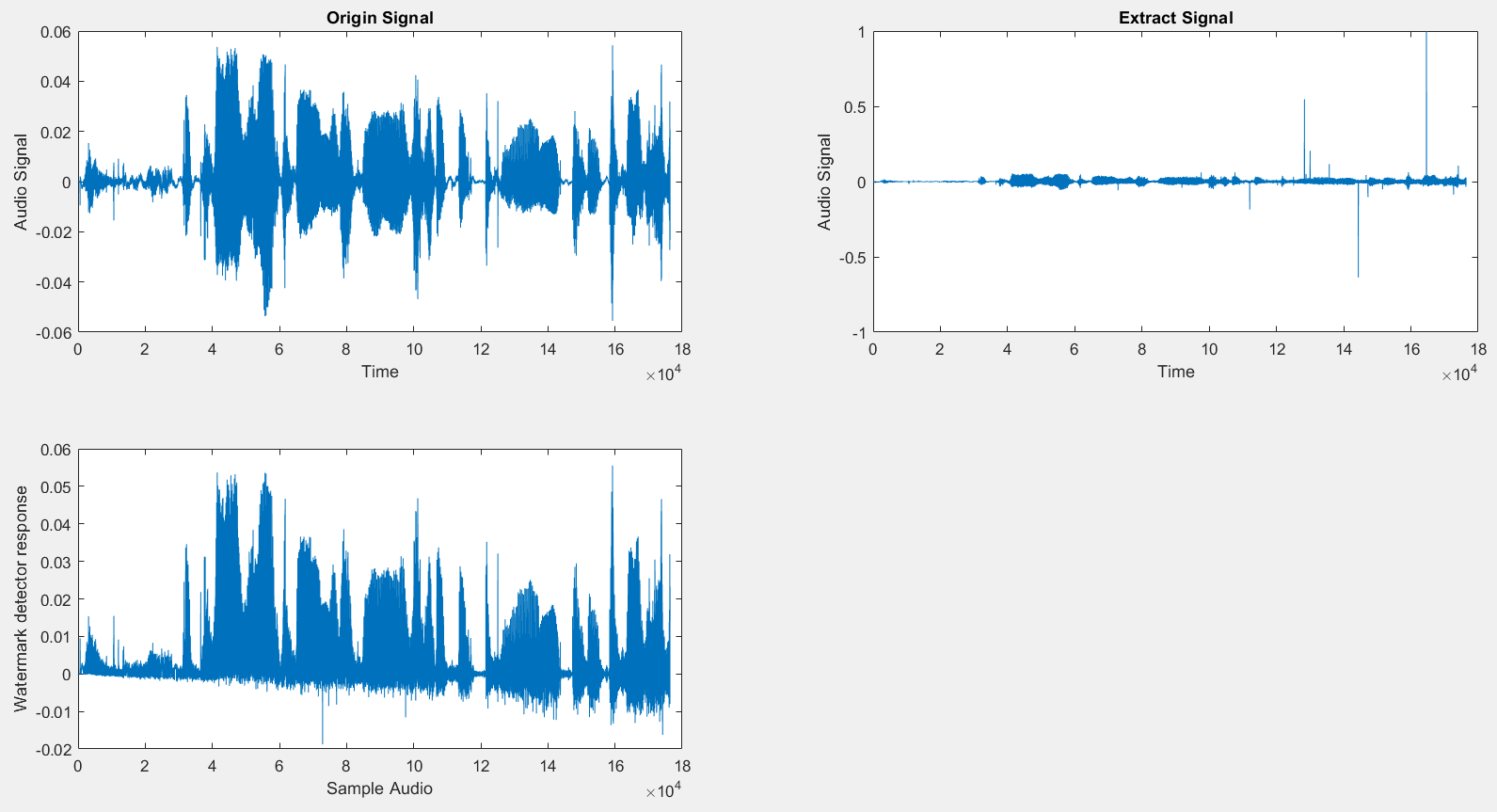
* Đọc hình ảnh gốc và âm thanh nhúng. Thực hiện DCT cho ra watermarked



* Thực hiện trích xuất signal message



* Thực hiện kiểm tra sự giống nhau của signal message và signal trích xuất

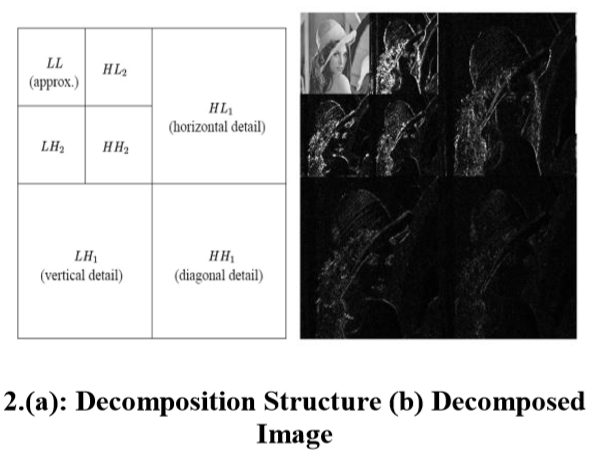


# Ẩn thông tin trong miền hybrid (Discrete Wavelet Transform)

## Mô tả thuật toán:

Thuật toán Discrete Wavelet Transform là thực hiện phân tách và tính toán tạo ra ma trận gần đúng và mạ trận hệ số chí tiết dựa trên các bộ lọc phân tách sóng con được chỉ định (hình 14). Các bộ lọc phân tích sóng gồm có Daubechies (‘db1’), Coiflets (‘coif1’), Symlets (‘sym2’), Discrete Meyer (‘dmey’), Biorthogonal (‘bior1.1’), and Reverse Biorthogonal (rbio1.1). Công thức của thuật toán DWT là

[Approximation, Horizontal, Vertical, Diagonal] = dwt2(A,'bior1.1');



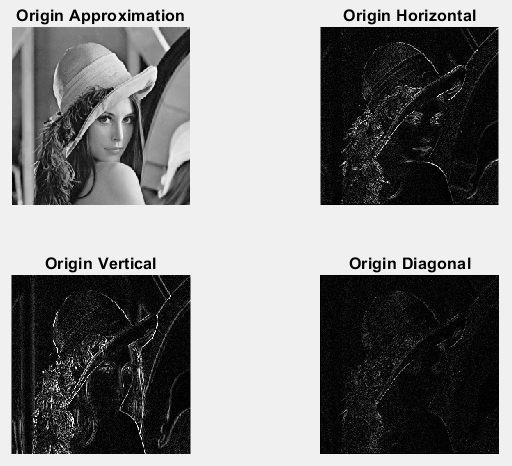
Hình 14. Thực hiện DWT với anh Lena

Quá trình nhúng của thuật toán DWT được đề xuất như sau:

Bước 1: Chọn hình ảnh gốc là hình ảnh Y học có kích thước 512x512

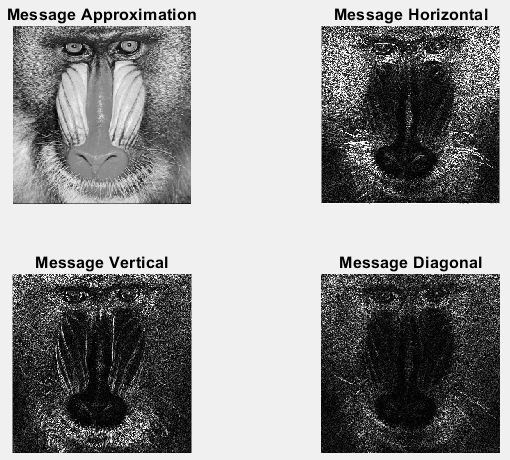
Bước 2: Chọn hình ảnh nhúng là hình ảnh Y học có kích thước 512x512

Bước 3: DWT với hình ảnh gốc ta thu được tần số thấp approximation (LL) và ba dải tần số cao vertical (LH), horizontal sub-band (HL) và diagonal sub-band (HH) với kích thước là 256x256 (hình 15).



Hinh 15. DWT ảnh Lena

Bước 5: DWT với hình ảnh nhúng ta thu được tần số thấp approximation (LL) và ba dải tần số cao vertical (LH), horizontal sub-band (HL) và diagonal sub-band (HH) với kích thước là 256x256 (hình 16).



Hình 16. DWT với Bamboo

Bước 6: Thực hình quá trình nhúng dữ liệu tại tần số thấp approximation.

Watermarked Approximation = Origin Approximation+(0.03 \* Message Approximation);

Bước 7: DWT nghịch đảo được áp dụng cho các dải con LL, LH, W và HH để thu được hình ảnh Watermarked.

watermarked = idwt2(Approximation, Horizontal, Vertical, Diagonal, 'bior1.1');

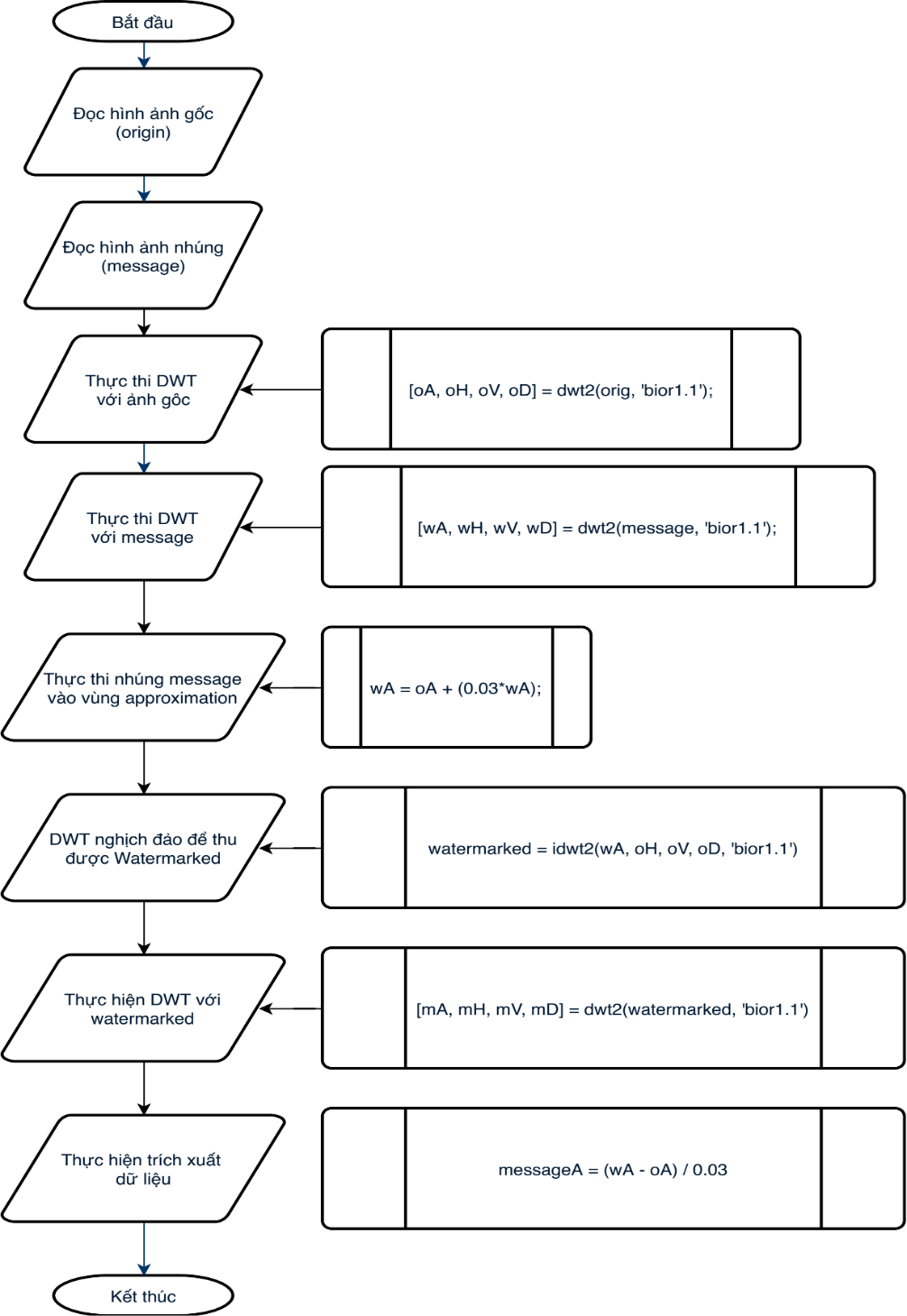
Bước 8: Thược hiện DWT với hình ảnh Watermarking ta thu được Approximation, Horizontal, Vertical, Diagonal.

Bước 9: Thước hiện trích xuất message:

Message Approximation = (Watermarked Approximation - Origin Approximation) / 0.03;

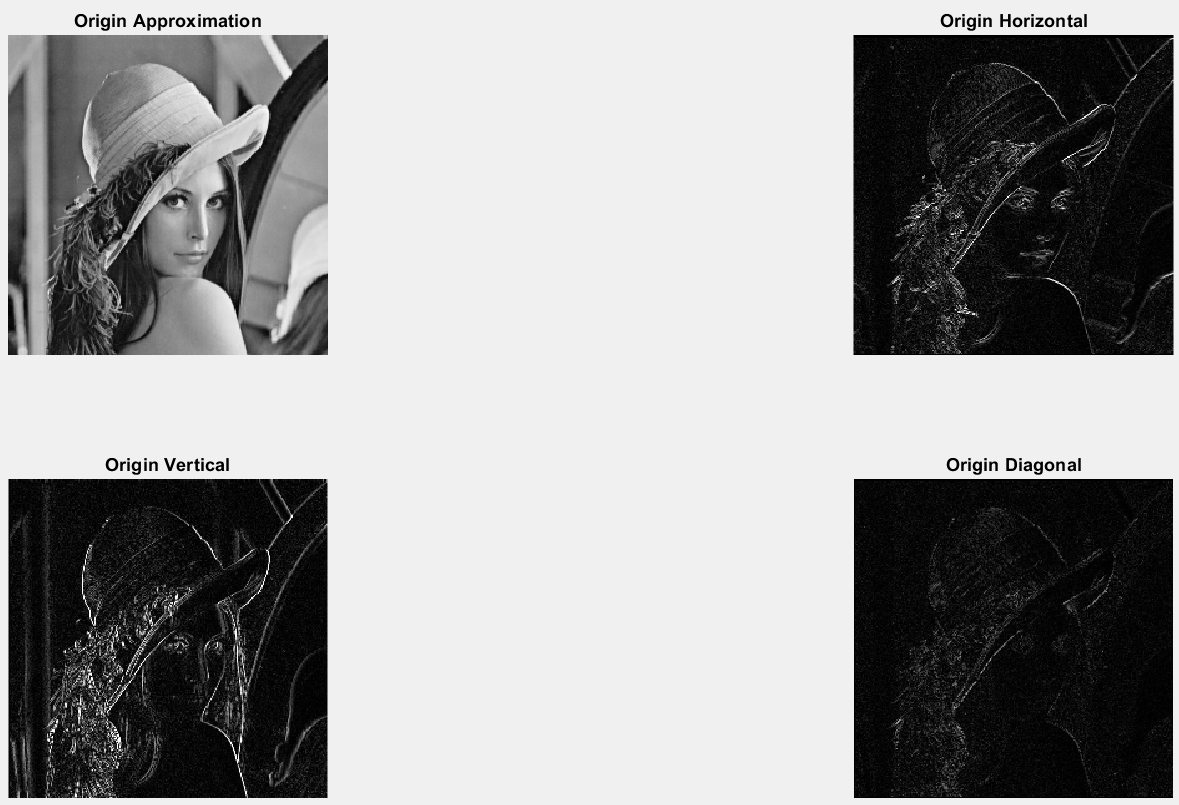
***Chú ý***: Thuật toàn này cũng có điểm giống DCT là một thuật toán với trích xuất bán mù, việc trích xuất thì có key.

## Sơ đồ quá trình nhúng:

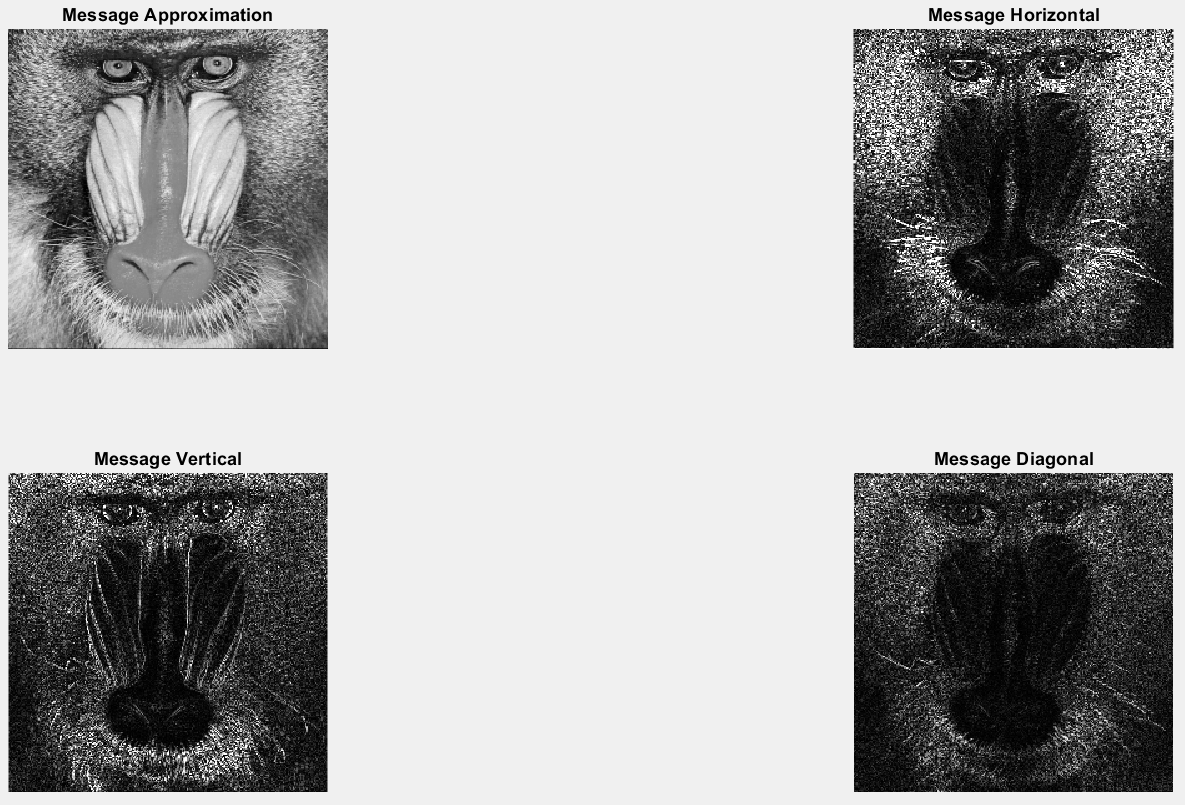


## Kết quả thực nghiệm

* Thực hiện DWT cho ảnh gốc với bộ lọc Biorthogonal.



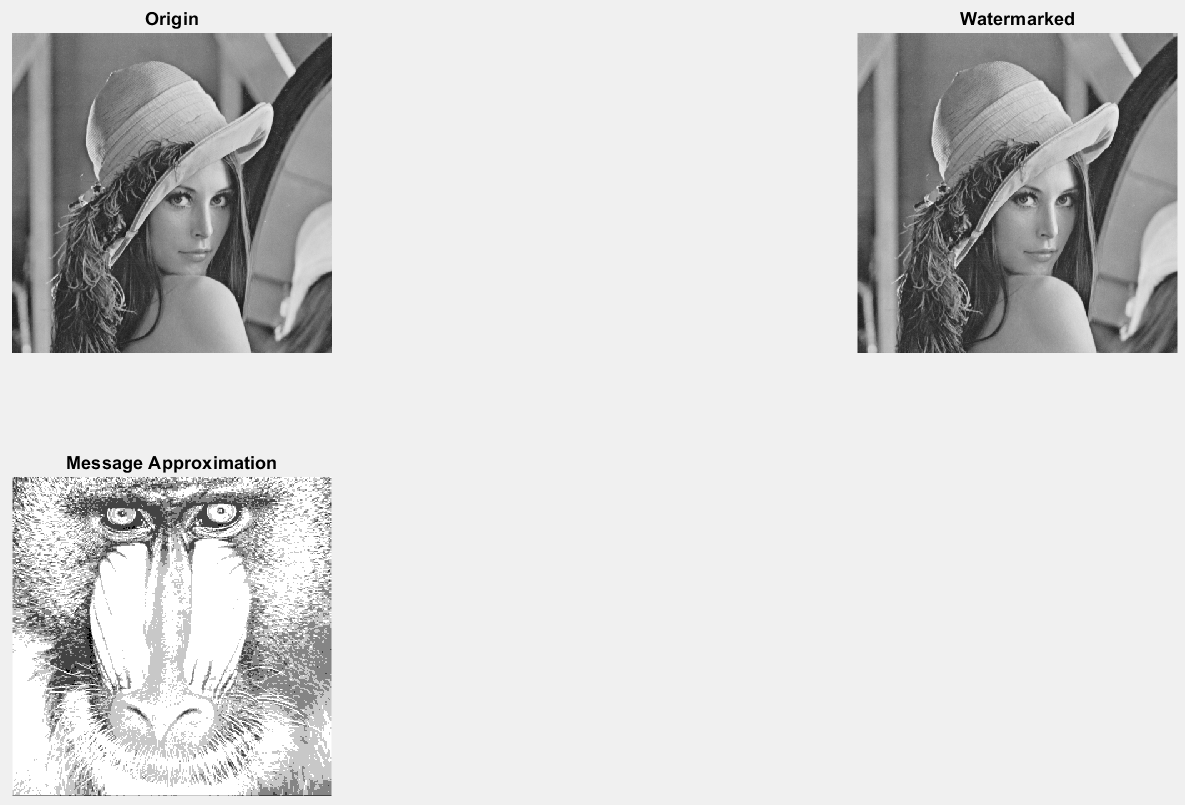
* Thực hiện DWT cho ảnh message với bộ lọc Biorthogonal.



* Kết quả thực hiện nhúng DWT



* Kết quả trích xuất message từ Watermarked



# Ẩn thông tin sử dụng thuật toán Normed Centre of Gravity (NCG) và Morphing

## Mô tả thuật toán:

Thuật nhúng dữ liệu được dựa trên việc morph image, bằng cách chia các hình thành các khối block rồi sau đó tìm trọng vector trọng tâm trong mỗi khối. Từ vector trọng tâm sẻ kiếm các đường cong có thể nhúng bit vô và dung morph image để biến đổi nó.

Từ mô tả trên ta có các bước thao tác của thuật toán như sau:

Bước 1: Đọc hình ảnh vật mang.

Bước 2: Phân tích message text sang bit

Bước 3: Chia hình thành các khối block

Bước 5: Xư lý trong từng block

+) Tính toán vector trọng tâm trong từng khối.

+) Tìm ra các đường cong có thể nhúng bit vô trong các khối.

+) Giảm một số hướng cong vênh được chọn gần nhau để giảm độ phức tạp tính toán cho các bước tiếp theo.

+) Tìm kiếm, lựa chọn các đường cong thô được nhúng bit vô trong khối block.

+) Lọc lại đường cong tốt nhất để xuất ra block mới.

Bước 10: Đọc hình watermarked

Bước 11: Chia watermarked ra từng thành từng block

Bước 12: Xư lý trong từng block

+) Tìm vector trọng tâm trong các block

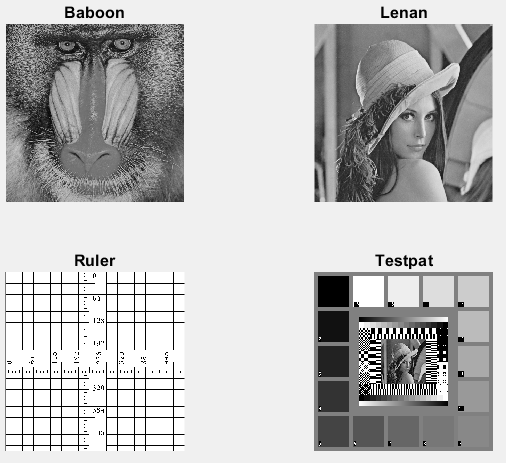
+) Compare độ sai khác làm tròn 0 hoặc 1 (kết quá bit nhúng)

## Sơ đồ quá trình nhúng:

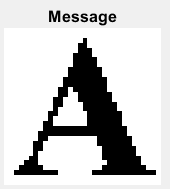


## Kết quả thực nghiệm

* Sẽ chạy thực nghiệm trên các hình ảnh vật mang sau (kích thước 512x512)



* Hình ảnh nhúng là ảnh nhị phân (32x32)



* Các tấn công sử dụng để kiểm chứng: nén JPEG, nhiễu, và một số biến đổi hình học. (Chỉ thu được kết của của block 4x4, 8x8. Còn block 32x32 và 64x64 thì không đủ dung lượng để nhúng)

|  |  |  |  |
| --- | --- | --- | --- |
| Baboon  (4x4) | Lena  (4x4) | Ruler  (4x4) | Testpat  (4x4) |
| PSNR: 31.5% | PSNR: 51.41% | PSNR: 22.47% | Lỗi |

Kết quả PSNR só sánh ảnh gốc và watermarked

|  |  |  |  |
| --- | --- | --- | --- |
| Loại tấn công | Baboon  (4x4) | Lena  (4x4) | Ruler  (4x4) |
| Nén JPED  QF=85% | NC: 98.13% | NC: 80.08% | NC: 62.66% |
| Nén JPED  QF=95% | NC: 98.98% | NC: 89.74% | NC: 45.36% |
| Noise 2%  “salt & pepper” | NC: 85.31% | NC: 74.48% | NC: 43.69% |
| Xoay 90 | Lỗi | Lỗi | Lỗi |
| Chuyển file TIFF sang JPED | NC: 100% | NC: 100% | NC: 43.43% |
| Crop 50% | Lỗi | Lỗi | Lỗi |
| Remove ngẫu nhiên | NC: 42.65% | NC: 42.65% | NC: 15.40% |

Kết quả NC so sánh vật nhúng và vật trích xuất

|  |  |  |  |
| --- | --- | --- | --- |
| Baboon  (8x8) | Lena  (8x8) | Ruler  (8x8) | Testpat  (8x8) |
| PSNR: 27.59% | PSNR: 31.17% | PSNR: 17.47% | Lỗi |

Kết quả PSNR só sánh ảnh gốc và watermarked

|  |  |  |  |
| --- | --- | --- | --- |
| Loại tấn công | Baboon  (8x8) | Lena  (8x8) | Ruler  (8x8) |
| Nén JPED  QF=85% | NC: 98.22% | NC: 94.20% | NC: 98.96% |
| Nén JPED  QF=95% | NC: 98.72% | NC: 98.01% | NC: 98.98% |
| Noise 2%  “salt & pepper” | NC: 74.23% | NC: 66.49% | NC: 84.66% |
| Xoay 90 | Lỗi | Lỗi | Lỗi |
| Chuyển file TIFF sang JPED | NC: 99.87% | NC: 100% | NC: 100% |
| Crop 50% | Lỗi | Lỗi | Lỗi |
| Remove ngẫu nhiên | NC: 99.74% | NC: 100% | NC: 100% |

Kết quả NC so sánh vật nhúng và vật trích xuất