



Università  
di Catania

# Progetto Multimedia

## Confronto tra algoritmi di Steganografia e Watermarking

Calderaro Graziana                    1000006198

Gallina Matteo                        1000055733

Gigliuto Emily                        1000007475

Docenti:                              Prof. Dario Allegra,

    Prof. Filippo Stanco

19 Marzo 2025

<b>Introduzione.....</b>	<b>4</b>
<b>Capitolo 2 - Algoritmi.....</b>	<b>5</b>
Least Significant Bit (LSB).....	5
Trasformata discreta del coseno (DCT).....	9
DCT Text.....	11
DCT SingleQuant.....	13
DCT Full.....	15
DCT JPEG like.....	17
DCT Low Frequency.....	19
Discrete Fourier Transform (DFT).....	22
Direct Sequence Spread Spectrum (DSS).....	25
Discrete Wavelet Transform (DWT).....	28
DWT One-Level.....	29
DWT Multi-Level.....	31
DWT - Otsu.....	33
<b>Capitolo 3 - Risultati.....</b>	<b>34</b>
Mean Squared Error - MSE.....	36
Peak Signal-to-Noise Ratio - PSNR.....	36
Structural Similarity Index - SSIM.....	37
LSB - Spatial.....	37
DCT - Text.....	43
DCT - SingleQuant.....	48
DCT - Full.....	52
DCT - JPEG - like.....	56
DCT - Low frequency.....	58
DFT.....	64
DSSS.....	68
DWT - OneLevel.....	72
DWT - MultiLevel.....	74
DWT - OTSU.....	78
<b>Capitolo 4 - Risultati a confronto.....</b>	<b>81</b>
LSB - Spatial.....	81
DCT - Single Quant.....	83
DCT - Full.....	87
DCT - Jpeg - Like.....	91
DCT - Low Frequency.....	95
DFT.....	98

DSSS.....	103
DWT - One - Level.....	107
DWT - MultiLevel.....	111
DWT - Otsu.....	115
<b>Conclusione.....</b>	<b>119</b>
<b>Riferimenti.....</b>	<b>120</b>

# Introduzione

L'obiettivo principale del progetto è quello di confrontare e sperimentare tecniche diverse sia nel campo della steganografia sia in quello del watermarking, due discipline fondamentali per la protezione dei contenuti digitali.

In un'epoca in cui la sicurezza delle informazioni è cruciale, queste metodologie offrono strumenti per garantire sia la riservatezza dei dati (nascondendo messaggi in maniera impercettibile) sia la tutela dei diritti d'autore e l'integrità dei contenuti (attraverso segni distintivi difficilmente rimovibili).

La steganografia si occupa dell'occultamento di dati all'interno di file multimediali senza alterare in modo percettibile il contenuto originale, mentre il watermarking mira a inserire segni distintivi all'interno di immagini o altri supporti digitali per garantire l'autenticità e la protezione contro la contraffazione o l'uso non autorizzato.

## L'approccio multidisciplinare del progetto

Il lavoro si è articolato nell'analisi e nell'implementazione di diverse tecniche, valutandone i punti di forza e le limitazioni. Lavorare in team ha permesso di unire competenze differenti e di confrontare metodologie che, pur condividendo obiettivi simili, adottano approcci tecnici molto diversi.

Questo confronto è stato fondamentale per valutare:

- Robustezza: Quanto il metodo scelto da ogni componente del gruppo resiste a operazioni di manipolazione o compressione.
- Impercettibilità: In che misura il metodo altera il contenuto originale senza evidenti artefatti.
- Efficienza computazionale: La complessità implementativa e il tempo di elaborazione richiesto.

# Capitolo 2 - Algoritmi

## Least Significant Bit (LSB)

LSB sta per Least Significant Bit, ovvero il bit meno significativo di un byte.

Nel contesto delle immagini digitali (es. PNG o BMP), ogni pixel è composto da uno o più canali (RGB), e ogni canale ha 8 bit (valori da 0 a 255). Il bit meno significativo è quello che influenza meno il valore complessivo del colore.

L'idea è sostituire i bit meno significativi dei pixel con i bit della “secret” (cioè l'informazione nascosta), senza alterare visibilmente l'immagine.

Nell'applicazione pratica si prende l'immagine cover (dove si nasconde il messaggio), si converte il messaggio segreto in binario, si sovrascrivono i bit meno significativi (LSB) dei pixel con i bit del messaggio e si ottiene un'immagine visivamente identica ma che contiene l'informazione nascosta.

Vantaggi:

- E' semplice da implementare.
- Nessuna perdita visiva.
- Funziona bene con immagini non compresse (es. BMP, PNG).

Svantaggi:

- Estremamente fragile: basta una compressione o un filtro per distruggere i dati.
- Facile da rilevare con strumenti di analisi statistica.
- Poca robustezza contro attacchi o alterazioni.

## Problema

L'algoritmo LSB nel dominio spaziale, pur garantendo una semplice implementazione e una buona capacità di embedding, risulta estremamente fragile quando l'immagine contenitore subisce anche minime trasformazioni. Operazioni comuni come la compressione con perdita (tipica del formato JPEG), il ridimensionamento, l'applicazione di filtri o anche un semplice ritaglio dell'immagine possono compromettere completamente l'integrità del messaggio nascosto.

Anche un'alterazione minima a livello di pixel può portare alla distruzione del messaggio, rendendo il metodo inadatto in scenari dove l'immagine può essere modificata, compressa o trasmessa in formati lossy.

In altri termini, l'LSB classico non offre alcuna robustezza contro manipolazioni, rendendolo vulnerabile e facilmente neutralizzabile. Per questo motivo, è necessario evolversi verso tecniche più resistenti, che sappiano garantire persistenza del dato nascosto anche in condizioni avverse.

Per aumentare la capacità di embedding, si può decidere di inserire la secret su più bit per ogni pixel (ad esempio 2 o 3 bit per canale). Tuttavia, ciò comporta un compromesso tra quantità di dati nascosti e visibilità dell'alterazione. La quantizzazione della secret in pochi bit consente un controllo più fine sulla qualità visiva, ma non offre protezione contro compressione, filtri o manipolazioni.

In sintesi, questa soluzione privilegia semplicità e capacità, ma risulta estremamente vulnerabile a qualsiasi tipo di trasformazione o degrado del file contenitore.

## Soluzione al problema

Per risolvere la fragilità intrinseca della steganografia LSB nel dominio spaziale, si adotta un approccio più sofisticato che opera nel dominio delle frequenze: la Trasformata Discreta del Coseno (DCT). Questo metodo, implementato nel file *DCT-text.py*, si ispira al principio della compressione JPEG, dove l'immagine viene scomposta in blocchi (es.  $8 \times 8$ ) e ogni blocco trasformato per evidenziare le sue componenti di frequenza.

Nel contesto steganografico, si selezionano specifici coefficienti DCT, tipicamente quelli a bassa-media frequenza, per inserire i bit del messaggio. Questi coefficienti presentano una maggiore stabilità in seguito a trasformazioni comuni, come la compressione o il filtraggio, rendendo l'embedding più robusto.

Inoltre, agendo in frequenza, si riduce la correlazione tra la modifica e la struttura spaziale dell'immagine, rendendo più difficile la rilevazione visiva o statistica del messaggio nascosto. Questo approccio rappresenta quindi una soluzione evolutiva rispetto all'LSB classico, bilanciando impercettibilità e resistenza.

## Trasformata discreta del coseno (DCT)

La DCT è una trasformazione matematica che converte un segnale (ad esempio, un'immagine) dallo spazio delle intensità (dominio spaziale) al dominio delle frequenze. In pratica, prende un insieme di valori (come i pixel di un'immagine) e li scomponete in una somma di coseni con diverse frequenze.

L'idea chiave è che le immagini reali hanno molta ridondanza: i pixel adiacenti tendono ad avere valori simili. La DCT decorrela i dati: ossia, li trasforma in coefficienti più indipendenti.

Dopo la trasformazione, la maggior parte delle informazioni visive importanti si concentra nelle basse frequenze.

L'uso della DCT nella steganografia nasce dall'esigenza di superare i limiti della LSB nel dominio spaziale, che è facilmente distruttibile. La DCT consente di nascondere i dati in componenti meno visibili all'occhio umano (es. frequenze medio-alte), aumentare la robustezza del messaggio contro compressione JPEG o filtri, sfruttare la ridondanza delle immagini in modo più intelligente, inserendo l'informazione dove è meno probabile venga alterata.

Vantaggi:

- Migliore robustezza rispetto all'LSB spaziale, soprattutto contro la compressione JPEG.
- Minor impatto visivo, se si scelgono coefficienti appropriati.
- Compatibilità con pipeline di compressione usate nel mondo reale (JPEG, video, ecc.).

Svantaggi:

- Capacità limitata: non si può inserire troppo testo, altrimenti si notano le alterazioni.
- Maggiore complessità computazionale rispetto all'LSB puro.
- Se l'algoritmo non è ben progettato, si rischia di perdere i dati nei coefficienti ad alta frequenza (che vengono compressi).
- È comunque vulnerabile a trasformazioni pesanti o cambiamenti di qualità elevati.

## DCT Text

L'algoritmo DCT-text rappresenta un passo intermedio tra la semplice steganografia spaziale (LSB) e approcci più avanzati basati su modifiche statistiche dei coefficienti. Qui l'idea principale è spostare l'embedding dal dominio dei pixel a quello delle frequenze, sfruttando la Discrete Cosine Transform (DCT).

L'immagine viene suddivisa in blocchi da  $8 \times 8$  pixel, come accade nella compressione JPEG. Ogni blocco viene trasformato con la DCT, ottenendo una matrice di coefficienti che rappresentano le frequenze spaziali dell'immagine.

Tra questi coefficienti, viene selezionato uno (solitamente ad alta frequenza, poco percepibile) e ne viene modificato il bit meno significativo (LSB) per inserire un bit del messaggio di testo.

Questo metodo conserva la semplicità logica della steganografia LSB, ma migliora leggermente la resistenza a modifiche dell'immagine, poiché l'alterazione avviene in una zona che ha minore impatto visivo.

## Problema

La tecnica LSB applicata nel dominio spaziale soffre di scarsa robustezza e non garantisce alcuna resistenza a compressione o trasformazioni. Per ovviare a questa debolezza, l'algoritmo DCT-text tenta un primo passo verso il dominio delle frequenze, ma presenta comunque due limiti principali:

1. La capacità di embedding è ridotta, in quanto ogni blocco  $8\times 8$  può veicolare al massimo un singolo bit di testo.
2. La modifica avviene spesso su coefficienti DCT ad alta frequenza, che tendono a essere eliminati dalla compressione JPEG, con conseguente fragilità residua.

Questo metodo rappresenta un compromesso: sfrutta la de-correlazione spaziale offerta dalla DCT per evitare di manipolare direttamente i pixel, ma continua a operare a livello di bit, mantenendo una logica semplice.

## Soluzione al problema

Per migliorare la robustezza e sfruttare più pienamente le proprietà statistiche dei coefficienti DCT, viene introdotta una quantizzazione mirata in un solo coefficiente selezionato, come implementato nel file *DCT-singleQuant.py*.

Questo approccio non si limita a modificare i bit, ma interviene su l'ampiezza numerica del coefficiente in modo controllato, codificando il messaggio attraverso la presenza o assenza di una determinata quantizzazione. Ne risulta una tecnica più robusta, capace di sopportare compressioni moderate, e in grado di mimetizzarsi meglio all'interno della struttura dell'immagine trasformata.

## DCT SingleQuant

L'algoritmo DCT-singleQuant rappresenta un'evoluzione mirata rispetto ai metodi che modificano direttamente i bit nei coefficienti DCT (come DCT-text). La sua principale innovazione consiste nel quantizzare in modo controllato un solo coefficiente per blocco anziché intervenire su più punti o agire direttamente a livello di bit. Questo approccio nasce dall'esigenza di aumentare la precisione dell'embedding e garantire un'estrazione del messaggio più affidabile, riducendo al minimo la distorsione visiva.

Come nei metodi basati sulla trasformata DCT, l'immagine viene prima suddivisa in blocchi da  $8 \times 8$  pixel, ciascuno dei quali viene trasformato nel dominio delle frequenze tramite la DCT. Questo processo produce una matrice di coefficienti che rappresentano le componenti di frequenza dell'immagine. A differenza del semplice LSB in frequenza, DCT-singleQuant sceglie un coefficiente stabile e ben posizionato, spesso a frequenza medio-bassa, per inserirvi un bit informativo.

Il principale vantaggio di questo approccio è la minima distorsione introdotta: poiché si lavora su un solo coefficiente per blocco, tutti gli altri restano invariati, preservando la struttura visiva dell'immagine. Questo si traduce in un'alta impercettibilità e in un'elevata fedeltà nella ricostruzione del messaggio nascosto.

## Problema

Nel passaggio dalla steganografia spaziale a quella basata su DCT, emerge una nuova problematica: le distorsioni involontarie. Infatti, quando si modifica un blocco di coefficienti DCT per inserire un bit, c'è il rischio che anche altri coefficienti (non target) vengano alterati indirettamente durante le operazioni di quantizzazione o ricostruzione dell'immagine.

Questo è particolarmente critico in scenari dove è richiesta un'estrazione perfetta del messaggio, ovvero dove ogni bit nascosto deve poter essere recuperato senza ambiguità, anche dopo operazioni di decompressione. La modifica simultanea di più coefficienti può introdurre rumore indesiderato e rendere l'estrazione meno affidabile.

Questo metodo migliora la qualità dell'immagine e aumenta l'affidabilità dell'estrazione, soprattutto in ambienti dove l'immagine può subire compressioni lievi o salvataggi multipli.

## Soluzione al problema

Nonostante l'accuratezza del metodo, DCT-singleQuant ha una capacità limitata, poiché utilizza solo un coefficiente per blocco. Questo può diventare un collo di bottiglia quando si vogliono trasmettere quantità maggiori di dati.

Per risolvere questo limite, si può adottare una strategia più aggressiva come quella implementata in *DCT-full.py*, che sfrutta tutti (o più) coefficienti del blocco DCT, aumentando notevolmente la capacità di embedding. Naturalmente, ciò richiede una gestione più attenta per evitare di compromettere la qualità visiva e la robustezza.

## DCT Full

L'algoritmo DCT-full nasce dall'esigenza di aumentare la capacità complessiva del sistema steganografico. In questo approccio, si adotta una strategia "full-band", ovvero si sfruttano tutti i coefficienti DCT all'interno di ciascun blocco per nascondere l'informazione, generalmente sotto forma di watermark.

L'algoritmo si distingue per la sua linearità e facilità di implementazione. L'uso della trasformata DCT su tutta l'immagine e l'inserimento additivo del watermark consentono di ottenere un metodo funzionale in poche righe di codice. Dal momento che il watermark viene distribuito su tutti i coefficienti DCT dell'immagine di copertura, l'intero contenuto del watermark può essere inserito, purché abbia la stessa dimensione dell'immagine host.

L'introduzione del parametro *alpha* consente di modulare la forza dell'inserimento, permettendo un compromesso tra invisibilità del watermark e qualità dell'estrazione. Valori bassi garantiscono alta impercettibilità; valori più alti migliorano la robustezza.

## Problema

L'alterazione di tutti i coefficienti DCT, inclusi quelli ad alta frequenza, rende il watermark estremamente vulnerabile. In particolare, le frequenze alte vengono solitamente eliminate o degradate in presenza di compressione lossy (es. JPEG), filtraggio (sfocature, edge enhancement) e trasformazioni geometriche (scalature, ritagli). Questo comporta una facile perdita o corruzione del watermark, anche con modifiche minime all'immagine watermarked.

L'algoritmo tratta tutti i coefficienti DCT con lo stesso peso, senza distinguere tra frequenze visivamente rilevanti e non. Questo approccio “piatto” ignora i modelli di percezione visiva che potrebbero guidare l'inserimento verso regioni meno sensibili, migliorando la qualità finale dell'immagine.

Il watermark da inserire deve avere le stesse dimensioni dell'immagine di copertura, a causa della corrispondenza uno-a-uno dei coefficienti DCT. Questo vincolo impone un ridimensionamento del watermark, con conseguente perdita di dettaglio e informazione.

## Soluzione al problema

Per ovviare a questo limite, si rende necessario un algoritmo che simuli il comportamento reale della compressione JPEG, sia in fase di embedding che di estrazione. *DCT-JPEG-like.py* nasce proprio con questo obiettivo: introdurre un embedding adattivo, che tenga conto della matrice di quantizzazione JPEG e operi in modo da preservare l'informazione nascosta anche dopo una reale compressione con perdita.

## DCT JPEG like

Questa variante dell'algoritmo implementa un embedding più realistico rispetto al processo di compressione JPEG, con l'obiettivo di inserire un watermark che simuli la pipeline reale della compressione.

A differenza della versione DCT-full, che opera sull'intera immagine in DCT, qui l'algoritmo suddivide l'immagine in blocchi  $8 \times 8$ , applica DCT locale a ciascun blocco, utilizza matrici di quantizzazione JPEG (Q50, Q90, Q100) e inserisce i bit del watermark direttamente nei LSB dei coefficienti quantizzati.

Il watermark è rappresentato da un'immagine di dimensioni  $32 \times 32$  (1024 pixel) convertita in 8192 bit, distribuiti su 4096 blocchi DCT con una capacità di 2 bit per blocco.

L'uso di blocchi  $8 \times 8$  e matrici di quantizzazione rende il metodo coerente con lo standard JPEG, migliorando la compatibilità e permettendo una simulazione realistica delle condizioni di compressione. Inserendo i bit nel dominio quantizzato, i dati risultano meno sensibili alla compressione successiva.

La capacità è regolabile tramite il parametro `bits_per_block`. È possibile adattare la dimensione del watermark e decidere quanti bit per blocco utilizzare, garantendo una buona scalabilità.

Il watermark non richiede l'adattamento completo alla dimensione della cover image, ma solo che la sua bitstream corrisponda alla capacità offerta dai blocchi.

## **Problema**

Sebbene simuli il flusso JPEG, l'inserimento avviene prima dell'effettiva compressione.

In caso di salvataggio reale in JPEG con qualità molto bassa, i LSB possono essere modificati o azzerati, compromettendo la fedeltà dell'estrazione.

L'inserimento nei coefficienti DCT quantizzati è sicuro solo se si scelgono frequenze medio-basse. In caso contrario (frequenze alte), il rischio di distorsione o eliminazione è maggiore.

Anche se minima, l'alterazione dei coefficienti può produrre piccole distorsioni visive nella stego image, specialmente in aree piatte o a basso contrasto.

L'algoritmo è più articolato e richiede una gestione accurata di bitstream, blocchi e quantizzazione/dequantizzazione.

## **Soluzione al problema**

Per garantire massima robustezza, l'evoluzione naturale è inserita nel file *DCT-lowFreq.py*, che concentra l'embedding nei coefficienti a bassa frequenza, notoriamente più stabili e resistenti anche alle compressioni più spinte. Operare su frequenze basse consente di sacrificare leggermente l'invisibilità, ma in cambio si ottiene una persistenza superiore del messaggio, anche in scenari critici.

## DCT Low Frequency

L'algoritmo DCT Low Frequency è progettato per risolvere un limite tipico dei metodi che operano in alta frequenza: la fragilità del messaggio nascosto quando l'immagine viene compressa, filtrata o ridimensionata. In queste situazioni, i coefficienti ad alta frequenza (dove molti algoritmi nascondono i dati per renderli invisibili) sono i primi a essere eliminati o modificati. La soluzione proposta da DCT-lowFrequency è semplice ma efficace: nascondere l'informazione nelle frequenze più basse della trasformata DCT.

L'immagine viene suddivisa in blocchi da  $8 \times 8$  pixel, su ciascun blocco viene applicata la Discrete Cosine Transform (DCT). Tra i coefficienti risultanti, vengono scelti solo quelli a bassa frequenza (quelli nella parte in alto a sinistra del blocco) per inserire i bit del messaggio. La modifica può avvenire tramite alterazione mirata del valore, quantizzazione personalizzata oppure semplici operazioni di parità (pari/dispari).

Le basse frequenze sono le più stabili e resistono bene a compressione JPEG, filtri di sfocatura, ridimensionamento e anche salvataggi multipli. Si ha alta affidabilità dell'estrazione: il messaggio può essere recuperato anche dopo modifiche consistenti all'immagine e vi è un ottimo compromesso tra invisibilità e durata: se calibrato bene, il watermark non è percepibile, ma è molto più difficile da rimuovere rispetto a uno in alta frequenza.

Ha una capacità inferiore per questo motivo si usano meno coefficienti per nascondere dati. Modificare le basse frequenze può avere un impatto leggermente più evidente sull'immagine, se il watermark è troppo forte.

Non è resistente a rotazioni o trasformazioni geometriche globali perché l'embedding avviene a livello di blocco.

## Problema

L'algoritmo DCT-lowFrequency nasce come risposta a una delle principali vulnerabilità dei metodi steganografici che operano nel dominio DCT: la sensibilità ai filtraggi locali (come il blur o il denoise) e alle operazioni di ri-campionamento, come il ridimensionamento o la ricampionatura dell'immagine.

In particolare, i metodi che nascondono dati nelle frequenze medie o alte tendono a perdere l'informazione steganografica quando l'immagine viene sottoposta a compressione o modifiche, perché proprio queste frequenze sono le prime ad essere compresse o eliminate. Questo rende il messaggio fragile, anche se visivamente invisibile.

## Soluzione del problema

Nonostante l'elevata stabilità offerta dall'uso delle basse frequenze, DCT-lowFrequency rimane comunque legato a un modello locale a blocchi. Questo significa che, in presenza di trasformazioni globali dell'immagine come rotazioni, scaling proporzionale o affini, oppure distorsioni geometriche, la struttura a blocchi viene spezzata o deformata, compromettendo l'estrazione del messaggio.

Per affrontare questa classe di problemi, è necessario uscire dal dominio dei blocchi locali e adottare una trasformazione che lavori su tutta l'immagine, mantenendo proprietà di invarianza geometrica. La soluzione è rappresentata dall'algoritmo *DFT.py*, che sfrutta la Discrete Fourier Transform, nota per la sua invarianza a traslazione, rotazione e scala rendendola particolarmente adatta a watermarking resistente in scenari reali e dinamici.

## Discrete Fourier Transform (DFT)

L'algoritmo DFT utilizza la Discrete Fourier Transform (DFT) per inserire un watermark in modo robusto alle trasformazioni geometriche globali, come rotazioni, traslazioni o ridimensionamenti. Mentre la DCT opera su blocchi (tipicamente 8x8), la DFT lavora sull'intera immagine, e grazie alla sua natura ciclica e complessa, offre proprietà di invarianza geometrica, risultando molto più robusta a trasformazioni globali.

Si calcola la DFT dell'intera immagine o di una porzione significativa. Il messaggio viene nascosto modificando la magnitudine (ampiezza) di alcuni coefficienti selezionati, evitando di toccare la fase. In seguito, viene eseguita la IDFT (Inverse DFT) per ricostruire l'immagine modificata. In fase di estrazione, si ricalcola la DFT e si confrontano le magnitudini per recuperare l'informazione.

### Vantaggi

- Resistente a trasformazioni geometriche globali: rotazioni, scalature, traslazioni.
- Difficile da rimuovere senza degradare pesantemente l'immagine.
- Adatto a scenari complessi (stampa, fotografia, social media...).

### Svantaggi

- Più complesso computazionalmente rispetto alla DCT.
- Minore compatibilità con la compressione JPEG.
- Le modifiche alla magnitudine possono essere più visibili se non calibrate bene.
- Difficile da sincronizzare in immagini tagliate o parzialmente corrotte.

## **Problema**

L'algoritmo DFT affronta il problema sfruttando la Discrete Fourier Transform (DFT), che permette di rappresentare l'intera immagine in termini di frequenze globali. In questo contesto, l'informazione viene nascosta modificando la magnitudine di coefficienti DFT selezionati (evitando di alterare la fase, che è più sensibile).

Dopo l'embedding, l'immagine viene ricostruita tramite trasformata inversa (IDFT).

Questa tecnica consente di distribuire il messaggio su tutta l'immagine, rendendolo più resistente a trasformazioni geometriche e meno vulnerabile a perdite localizzate.

## **Soluzione al problema**

Nonostante l'uso della DFT offra una certa invarianza geometrica, resta comunque sensibile a rumore casuale, compressioni aggressive o attacchi intenzionali.

Per migliorare ulteriormente la robustezza globale, si può adottare un approccio basato sul Direct Sequence Spread Spectrum (DSSS), come in *DSSS.py*.

Questa tecnica distribuisce ogni bit del messaggio su un'ampia banda di frequenze, mascherandolo come rumore e rendendo estremamente difficile la sua distruzione o rilevazione, anche in presenza di disturbi o attacchi attivi.

## Direct Sequence Spread Spectrum (DSS)

Il DSS è una tecnica di comunicazione nata per rendere i segnali più robusti e difficili da rilevare, utilizzando una sequenza pseudo-casuale per diffondere ogni bit su una larga banda. Nella steganografia, questa tecnica viene adattata per nascondere un messaggio su molti coefficienti dell'immagine (es. nel dominio DCT), mascherandolo come rumore.

Mentre la DFT nasconde i dati in posizioni frequenziali precise, modificando la magnitudine dei coefficienti globali dell'immagine, il DSS invece distribuisce il messaggio su molti coefficienti (es. tutti i blocchi DCT) seguendo una sequenza pseudo-casuale, rendendo il messaggio più resistente ai danni e meno riconoscibile.

Si genera una sequenza pseudo-casuale nota (la chiave). Ogni bit del messaggio viene moltiplicato per questa sequenza e sparso su più coefficienti (es. DCT). In fase di estrazione, si correla la sequenza dell'immagine con la chiave per ricostruire il messaggio originale.

### Vantaggi

- Alta robustezza contro rumore, filtri, attacchi casuali.
- Difficile da rilevare: sembra rumore distribuito.
- Possibilità di recuperare il messaggio anche se parzialmente danneggiato.

### Svantaggi

- Minor localizzazione: si lavora solo in frequenza, senza ancoraggio spaziale.
- Bassa capacità: i messaggi devono essere piccoli per restare impercettibili.
- Complessità maggiore nella sincronizzazione tra embedding ed estrazione.

## **Problema**

Anche le tecniche steganografiche che operano nel dominio delle frequenze, come quelle basate su DCT o DFT, possono risultare vulnerabili a rumore casuale, compressioni non prevedibili o manipolazioni localizzate dei coefficienti, ad esempio tramite filtri, attacchi statistici o modifiche mirate. Quando il messaggio è codificato in punti specifici (es. un coefficiente per blocco), è sufficiente alterare quei punti per distruggerlo o corromperlo.

Serve quindi un approccio che distribuisca il messaggio in modo più ampio e resistente.

## **Soluzione al problema**

Sebbene il DSSS aumenti notevolmente la robustezza contro attacchi casuali, il suo punto debole resta la scarsa localizzazione spaziale: si lavora sempre nel dominio della frequenza, perdendo l'ancoraggio preciso ai dettagli visivi dell'immagine.

Per superare questa limitazione, si può passare a una tecnica che combina dominio spaziale e frequenziale, come quella implementata in *DWT-oneLevel.py*, basata sulla Discrete Wavelet Transform (DWT). La DWT consente una scomposizione multirisoluzione che conserva sia la localizzazione spaziale che la distribuzione in frequenza, offrendo un compromesso molto potente tra capacità, invisibilità e robustezza.

## Discrete Wavelet Transform (DWT)

La DWT è una trasformazione matematica che, a differenza di DCT o DFT, analizza un segnale (o un'immagine) sia nel dominio della frequenza che in quello spaziale. Divide l'immagine in sottobande a diversi livelli di dettaglio, mantenendo informazioni localizzate (dove e quanto cambia un'immagine).

In pratica, divide l'immagine in zone che contengono: le forme generali (cioè le basse frequenze) e i dettagli come bordi e variazioni (cioè le alte frequenze)

### Vantaggi

- Combina analisi spaziale e frequenziale: utile contro attacchi locali.
- Supporta multi-risoluzione: si può inserire l'informazione a diversi livelli di dettaglio.
- Più resistente di LSB: puro e meno visibile se usata in sottobande corrette.

### Svantaggi

- Più complessa rispetto a LSB e DCT.
- Capacità variabile in base al livello di decomposizione.
- La scelta della sottobanda e del livello di embedding va calibrata con attenzione.

## DWT One-Level

La DWT-oneLevel è un'applicazione semplice della DWT, che utilizza una scomposizione a un solo livello (one-level) dell'immagine tramite wavelet Haar.

DWT-oneLevel lavora su blocchi localizzati con frequenze reali mantenendo posizione e contesto dell'informazione. DSS lavora su tutti i coefficienti, spargendo l'informazione in modo globale e pseudo-casuale: massima robustezza, ma senza localizzazione.

L'algoritmo funziona applicando una singola trasformazione wavelet sull'immagine semplice e veloce da calcolare. Quando questa trasformazione viene eseguita, l'immagine viene scomposta in quattro aree, chiamate sottobande: una che contiene l'approssimazione dell'immagine (cioè le informazioni più stabili e a bassa frequenza), e tre che contengono i dettagli orizzontali, verticali e diagonali (che rappresentano i cambiamenti più marcati nei diversi orientamenti).

In questo algoritmo, si sceglie di nascondere il messaggio nella sottobanda orizzontale, detta cH, perché contiene dettagli utili ma non troppo sensibili, e offre un buon equilibrio tra invisibilità e resistenza. I bit del messaggio vengono inseriti nei bit meno significativi dei coefficienti di cH, in modo da non alterare visibilmente l'immagine. Una volta modificata la sottobanda, si esegue l>IDWT (inverse DWT) per ricostruire l'immagine completa, che risulterà visivamente simile all'originale ma con il messaggio nascosto al suo interno.

## Problema

Nei metodi tradizionali, esiste un compromesso difficile tra robustezza (tipica dei metodi che lavorano in frequenza) e localizzazione spaziale (tipica dei metodi che agiscono su pixel o blocchi specifici). I metodi basati su DCT o DFT, pur offrendo una buona protezione alle trasformazioni o al rumore, perdono completamente il riferimento spaziale: non si sa dove, nell'immagine, è stata inserita l'informazione. Questo può rendere fragile l'embedding in presenza di manipolazioni parziali o localizzate.

Utilizzando una wavelet Haar, che consente di decomporre l'immagine in sottobande:

- cA (approssimazione, bassa frequenza),
- cH (dettagli orizzontali),
- cV (verticali),
- cD (diagonali).

## Soluzione al problema

L'algoritmo DWT one-level consente di nascondere dati in una singola scomposizione dell'immagine, ma questo approccio sfrutta solo in parte il potenziale della trasformata wavelet.

Per migliorare ulteriormente capacità e resilienza, si può adottare un modello multi-livello, come in *DWT-multiLevel.py*, che consente di nascondere informazioni a diverse scale di dettaglio, aumentare il numero di sottobande disponibili per l'embedding e rendere il messaggio più ridondante e persistente anche in scenari degradati.

## DWT Multi-Level

La DWT multi-level è un'estensione della trasformazione wavelet a singolo livello: invece di fermarsi alla prima scomposizione dell'immagine, il processo viene ripetuto più volte in modo gerarchico. Dopo il primo livello, la trasformata viene applicata nuovamente solo alla sottobanda cA (quella che contiene l'approssimazione dell'immagine), continuando così a suddividerla in nuovi insiemi di sottobande sempre più piccoli e più astratti.

Questo approccio consente di ottenere una decomposizione a più risoluzioni, in cui ogni livello fornisce informazioni diverse: i primi contengono dettagli fini e localizzati, mentre quelli più profondi conservano la struttura generale dell'immagine.

Nel contesto della steganografia, la DWT multi-level è utile perché permette di distribuire il messaggio su più livelli di dettaglio, aumentando sia la capacità che la robustezza, e offrendo anche più flessibilità nella scelta di dove e come nascondere le informazioni.

## Problema

L'algoritmo DWT-multiLevel risolve il limite di nascondere dati in una singola scomposizione dell'immagine eseguendo una decomposizione wavelet a più livelli: la DWT viene applicata ricorsivamente alla sottobanda cA, generando una gerarchia di sottobande di dettaglio (orizzontale, verticale, diagonale) a diverse scale.

In questo modo, si ottiene una visione completa dell'immagine, dalla struttura globale ai minimi dettagli. Il messaggio viene poi nascosto usando una tecnica LSB distribuita in tutte le sottobande di dettaglio, migliorando così sia la capacità totale di embedding, sia la robustezza rispetto a manipolazioni che colpiscono solo alcune porzioni dell'immagine.

Questa diffusione su più livelli e scale consente una maggiore flessibilità: si può scegliere se privilegiare invisibilità (livelli alti), resistenza (livelli profondi), o bilanciare entrambe.

## Soluzione al problema

Nonostante l'aumento di capacità e profondità, l'estrazione del watermark può diventare più complessa, soprattutto se si tratta di informazioni strutturate come maschere binarie (es. loghi o QR code). In questi casi, per garantire un'estrazione più precisa e visivamente netta, è possibile utilizzare un'ulteriore tecnica basata sulla segmentazione automatica, come quella implementata in *DWT-otsu.py*, che applica la soglia di Otsu per separare con maggiore affidabilità il contenuto utile dallo sfondo.

## DWT - Otsu

DWT - Otsu è un algoritmo di steganografia avanzata pensato per nascondere e recuperare immagini binarie (come loghi o QR code) in modo pulito e robusto, usando una combinazione tra trasformata wavelet (DWT) e tecniche di elaborazione dell'immagine.

Nasce per risolvere il problema dell'estrazione rumorosa, che si verifica spesso quando si nasconde un'immagine semplice (in bianco e nero) dentro un'immagine complessa, e poi si prova a recuperarla.

L'algoritmo lavora in due fasi: embedding e estrazione.

1. *Fase di embedding (inserimento)*: L'immagine contenente il messaggio (es. una maschera binaria) viene nascosta all'interno dell'immagine host attraverso una DWT multi-livello. I bit vengono distribuiti in diverse sottobande di dettaglio (orizzontale, verticale, diagonale) per garantire robustezza e capacità.
2. *Fase di estrazione (recupero)*: Dopo aver ottenuto l'immagine modificata, si esegue la DWT inversa per recuperare il watermark grezzo, che però spesso risulta sfocato o degradato.

Per migliorare la qualità, si applica:

- la soglia di Otsu, un algoritmo che trova in automatico il punto ideale per binarizzare l'immagine (cioè separare bianco e nero);
- un'ulteriore pulizia morfologica, che corregge piccoli artefatti, pixel isolati o rumore residuo, migliorando il risultato finale.

Questo approccio è perfetto quando si vuole nascondere un'immagine binaria in modo robusto e recuperarla senza ambiguità. Grazie all'uso di Otsu e della morfologia, il risultato finale è netto, leggibile e privo di distorsioni visive, anche se l'immagine host è stata compressa o alterata.

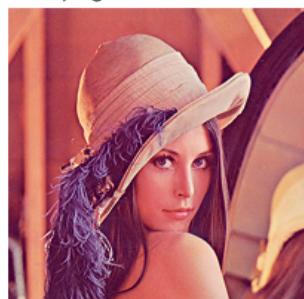
# Capitolo 3 - Risultati

Al fine di verificare le performance dei vari algoritmi implementati, sono state utilizzate una serie di immagini di test contenute nella cartella `../images/`.

Tutte le immagini sono in formato PNG e, per garantire uniformità nell'analisi, sono state preventivamente convertite in scala di grigi.

Per ciascun esperimento, è stato inoltre implementato un sistema interno di adattamento automatico tra la dimensione del messaggio segreto e la capacità dell'immagine da steganografare o watermarkare. Questo consente di evitare errori di overflow e garantisce che il messaggio venga inserito integralmente o ridimensionato in modo proporzionale alla disponibilità offerta dal supporto visivo.

lena.png 512×512



unicorn.png 512×512



pattern.png 512×512



logo.png 512×512



logoUni.png 1200×592



Università  
di Catania

Per valutare la bontà degli algoritmi si è scelto di utilizzare le seguenti metriche:

### Mean Squared Error - MSE

L'Errore Quadratico Medio è una metrica che quantifica la differenza media al quadrato tra i pixel dell'immagine originale e quelli dell'immagine alterata (es. dopo steganografia o watermarking). Più il valore di MSE è basso, più le due immagini sono simili. Un MSE pari a 0 indica immagini identiche.

$$MSE = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N [I(i,j) - K(i,j)]^2$$

- $M, N$ : dimensioni dell'immagine (altezza e larghezza)
- $I(i,j)$ : valore del pixel dell'immagine originale
- $K(i,j)$ : valore del pixel dell'immagine alterata

### Peak Signal-to-Noise Ratio - PSNR

Il Peak Signal-to-Noise Ratio misura la qualità di una immagine alterata rispetto all'originale, basandosi sul valore di MSE. È espresso in decibel (dB). Valori più alti di PSNR indicano una maggiore somiglianza.

Generalmente un PSNR:

- ★ > 40 dB → differenze impercettibili
- ★ 30 – 40 dB → buona qualità
- ★ < 30 dB → degrado visibile

$$PSNR = 10 \cdot \log_{10} \left( \frac{MAX_I^2}{MSE} \right)$$

oppure

$$PSNR = 20 \cdot \log_{10} \left( \frac{MAX_I}{\sqrt{MSE}} \right)$$

- $MAX_I$ : valore massimo possibile di un pixel (255 per immagini a 8 bit)
- $MSE$ : errore quadratico medio calcolato in precedenza

## Structural Similarity Index - SSIM

Lo Structural Similarity Index confronta due immagini valutando luminanza, contrasto e struttura. A differenza di MSE e PSNR, tiene conto della percezione visiva umana.

Il valore di SSIM è compreso tra 0 e 1:

- ★ SSIM = 1 → immagini perfettamente identiche
- ★ Valori più vicini a 1 → alta similarità strutturale

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + C_1)(\mu_y^2 + C_1)}$$

- $\mu_x, \mu_y$ : media dei valori dei pixel delle due immagini
- $\sigma_x^2, \sigma_y^2$ : varianza
- $\sigma_{xy}$ : covarianza
- $C_1, C_2$ : costanti di stabilizzazione (tipicamente piccole)

## LSB - Spatial

Questo approccio è noto per la sua estrema semplicità ed efficienza, ed è particolarmente adatto a scopi didattici o dimostrativi. L'immagine mostra i risultati di un algoritmo di watermarking tramite Lsb-Spatial:

Original vs. Stego Image MSE: 3.23, PSNR: 43.04 dB, SSIM: 0.9911  
Original vs. Extracted Secret MSE: 0.00, PSNR: inf dB, SSIM: 1.0000  
Embedding Time: 0.00353s, Extraction Time: 0.00251s

## Cover Image



### Original Secret Image

## Stego Image



Extracted Secret Image



Questa immagine mostra i risultati dell'algoritmo di steganografia (o watermarking) su una coppia di immagini: la cover image (in alto a sinistra) e la stego image (in alto a destra), con il messaggio segreto (in basso) rappresentato dal logo "GEM".

L'inserimento del watermark è stato eseguito con un sistema di adattamento automatico che ridimensiona il messaggio segreto in base alla capacità disponibile dell'immagine, evitando così errori o troncamenti.

Inoltre, non sono presenti limitazioni sulla dimensione dell'immagine: l'algoritmo è stato progettato per operare su immagini di qualsiasi risoluzione, adattando dinamicamente la quantità e il posizionamento dei dati nascosti.

### ★ **MSE = 3.23**

Dopo l'inserimento del logo "GEM" all'interno della cover image, il valore di MSE tra l'immagine originale e quella stego è aumentato a 3.23. Questo avviene perché l'inserimento del watermark modifica i bit meno significativi di alcuni pixel. Anche se le alterazioni sono minime, il MSE tiene conto di ogni differenza quadratica tra i pixel, quindi è sensibile anche a piccole variazioni sparse su tutta l'immagine.

### ★ **PSNR = 43.04 dB**

Il valore di PSNR è pari a 43.04 dB, un valore elevato che indica una qualità visiva praticamente indistinguibile. Ciò accade perché il rapporto segnale/rumore è molto favorevole: le modifiche apportate sono leggere e ben distribuite, quindi non degradano visibilmente la qualità complessiva dell'immagine.

### ★ **SSIM = 0.9911**

Il valore di SSIM è 0.9911, molto vicino a 1, il che conferma che la struttura visiva dell'immagine non è stata alterata in modo percepibile. Questo è coerente con l'idea che l'occhio umano non noti le variazioni introdotte dal watermark, soprattutto se queste non alterano contorni o pattern importanti.

### ★ **MSE = 0 / SSIM = 1 per l'immagine segreta**

Il confronto tra l'immagine segreta originale e quella estratta mostra valori di:

**MSE = 0**

**PSNR =  $\infty$  (infinito)**

**SSIM = 1.0000**

Significa che il logo è stato recuperato perfettamente, senza alcuna perdita informativa o distorsione. Il processo di embedding ed estrazione è quindi perfettamente reversibile per questo esempio.

## ★ Tempi

**Embedding Time: 0.00353s**

**Extraction Time: 0.00251s**

Indicano un'ottima efficienza computazionale, adatta anche per sistemi in tempo reale o embedded.

I risultati ottenuti evidenziano l'efficacia dell'algoritmo implementato per l'inserimento del logo "GEM" tramite tecniche di steganografia. L'analisi delle metriche oggettive mostra una modifica minima sull'immagine originale ( $MSE = 3.23$ ,  $PSNR = 43.04$  dB,  $SSIM = 0.9911$ ), confermando l'impercettibilità del watermark.

Al contempo, l'immagine segreta è stata recuperata integralmente e senza perdita ( $MSE = 0$ ,  $SSIM = 1$ ), dimostrando l'affidabilità e la precisione del processo di codifica e decodifica.

Il ridimensionamento automatico del messaggio e l'assenza di vincoli sulla dimensione dell'immagine rendono l'approccio flessibile e adattabile a diversi scenari applicativi, anche in tempo reale grazie ai tempi di elaborazione estremamente ridotti.

## DCT - Text

L'estrazione del DCT- Text avviene analizzando gli stessi coefficienti, recuperando i bit e ricostruendo il testo. La quantizzazione consente un controllo sulla distorsione, mentre la scelta del coefficiente alto riduce la percepibilità del watermark.

Il messaggio testuale viene ricostruito senza errori, anche se non è possibile calcolare MSE/PSNR come per le immagini, ma questo conferma l'efficacia dell'approccio. Questo è un compromesso efficace tra qualità visiva, sicurezza e semplicità, ideale per l'inserimento di messaggi brevi in immagini ad alta fedeltà.

Il metodo produce una stego image praticamente indistinguibile dall'originale e testo nascosto leggibile al 100%, rendendolo ideale per comunicazione nascosta o firma digitale leggera.

Cover vs. Stego MSE: 6.05, PSNR: 40.31 dB, SSIM: 0.9712  
Embedding Time: 1.81031s, Extraction Time: 0.94892s



Insert stego text:

Progetto Multimedia

secret message: Progetto Multimedia

In questo esperimento, è stato inserito un messaggio testuale ("Progetto Multimedia") all'interno della cover image tramite una tecnica basata su trasformata DCT. L'output mostra sia la qualità della stego image, sia la corretta estrazione del messaggio nascosto. Questi valori indicano una buona qualità visiva: il watermark è poco percepibile, ma non completamente invisibile, con una leggera distorsione causata dall'inserimento del testo nella frequenza.

### ★ **MSE = 6.05**

Un MSE del valore di 6.05 indica che sono state apportate modifiche più marcate rispetto a un embedding leggero (come LSB), a causa delle alterazioni sui coefficienti DCT.

L'aumento dell'MSE è logico: più intensa è la modifica della frequenza, maggiore è la variazione nei pixel, soprattutto in aree con bassa variazione cromatica.

### ★ **PSNR = 40.31 dB**

$$PSNR = 10 \log_{10} \left( \frac{255^2}{MSE} \right)$$

Un PSNR con risultato 40.31 dB è un valore ancora piuttosto alto, indicando buona qualità visiva.

Tuttavia, è inferiore a PSNR di embedding LSB tipico (>43 dB), riflettendo l'impatto visivo leggermente percepibile dovuto al DCT embedding per testo.

### ★ **SSIM = 0.9712**

L'SSIM con risultato 0.9712 indica un'ottima preservazione dell'aspetto visivo complessivo.

Pur presente una modifica a livello di alta frequenza, la struttura percepibile dell'immagine è maggiormente conservata, tranne nelle zone più piatte o uniformi.

### ★ **Testo Estratto: "Progetto Multimedia"**

Il messaggio è stato recuperato correttamente e integralmente, confermando l'efficacia dell'algoritmo nella codifica di testo nei coefficienti DCT.

L'algoritmo DCT, che interviene direttamente sui coefficienti di frequenza dei blocchi  $8 \times 8$ , presenta alcune limitazioni in contesti specifici. In particolare, può generare artefatti visivi evidenti in due casi:

- Zone uniformi o piatte: anche minime alterazioni in queste aree risultano facilmente percepibili, rendendo il watermark visivamente intrusivo.
- Pattern regolari o ripetuti: l'inserimento del watermark può rompere la coerenza visiva del pattern, con effetti disturbanti per l'occhio umano.

L'impiego di tecniche di dithering o perturbazioni controllate può aiutare a distribuire il watermark in modo più naturale, integrandolo meglio nel rumore visivo dell'immagine. Questo approccio contribuisce a ridurre la percepibilità, soprattutto in aree a bassa complessità visiva.

Per ridurre l'impatto visivo e aumentare la robustezza, è preferibile evitare i coefficienti a bassa frequenza, privilegiando invece quelli a media frequenza. Questi offrono un buon compromesso: sono meno evidenti all'occhio umano e al tempo stesso più resistenti a compressioni e manipolazioni dell'immagine.

Una possibile evoluzione consiste nello sviluppo di un sistema di embedding adattivo, capace di regolare l'intensità della modifica in base alla complessità locale dell'immagine. In zone dettagliate si può osare di più, mentre in aree piatte è utile limitare l'intervento per non generare artefatti visibili.

L'esperimento dimostra che l'inserimento di un watermark testuale tramite DCT è efficace dal punto di vista della decodifica. Tuttavia, emergono alcune criticità visive legate alla natura dell'immagine host. Per applicazioni più complesse o immagini particolarmente sensibili, è necessario un maggiore equilibrio tra invisibilità, robustezza e adattamento al contenuto visivo.

## DCT - SingleQuant

L'estrazione del DCT-singleQuant si basa sulla lettura di un coefficiente specifico per blocco, la cui quantizzazione controllata permette di distinguere i bit del messaggio. Agendo solo su un valore per blocco (tipicamente  $8 \times 8$ ), si riduce la distorsione complessiva. La selezione di coefficienti stabili garantisce una buona precisione nell'estrazione.

Nel contesto dell'embedding DCT, l'immagine host viene divisa in blocchi  $8 \times 8$ , e l'informazione viene inserita nei coefficienti di ciascun blocco.

Per una cover image di  $512 \times 512$  pixel, si ottengono esattamente:

$$\frac{512}{8} X \frac{512}{8} = 64 X 64 = 4096 \text{ blocchi}$$

L'algoritmo prevede di usare solo una parte di questi blocchi per aumentare la robustezza o evitare visibilità, è normale limitare la dimensione dell'immagine segreta a, ad esempio,  $32 \times 32 = 1024$  bit, perfettamente gestibili con una strategia selettiva su una porzione di blocchi.

Inoltre, un'immagine segreta piccola consente un inserimento più sicuro e meno invasivo, adatto a testare la precisione di estrazione del metodo.

Cover vs Stego: MSE = 56.48, PSNR = 30.61 dB  
Secret vs Extracted: MSE = 0.00, PSNR = inf dB  
Embedding Time: 0.88 s, Extraction Time: 0.43 s

Cover Image (512x512)



Original Secret Image (32x32)



Stego Image (DCT embedding)



Extracted Secret Image



## ★ **MSE = 56.48**

Il valore di MSE è più alto rispetto al metodo LSB (56.48 contro 3.23), indicando una maggiore alterazione dei pixel tra l'immagine originale e quella stego.

Questo è coerente con l'approccio DCT, che modifica i coefficienti frequenziali dei blocchi dell'immagine e poi ricostruisce i pixel, introducendo inevitabilmente distorsioni più evidenti, soprattutto se il watermark viene inserito in frequenze più basse.

## ★ **PSNR = 30.61 dB**

Un PSNR di 30.61 dB indica che le alterazioni sono percepibili, soprattutto in zone uniformi o dettagliate.

È comunque un valore accettabile in contesti dove l'impercepibilità perfetta non è prioritaria, ad esempio in applicazioni in cui è più importante la robustezza del watermark (come nel copyright digitale).

## ★ **Immagine Segreta – Recupero Perfetto**

**MSE = 0**

**PSNR =  $\infty$  (infinito)**

Il logo "GEM" (di dimensioni  $32 \times 32$ ) è stato estratto senza alcuna perdita, nonostante l'alterazione maggiore della cover image. Questo dimostra che il metodo DCT è estremamente efficace per la robustezza del watermark.

## ★ **Tempi di Elaborazione**

**Embedding Time: 0.88 secondi**

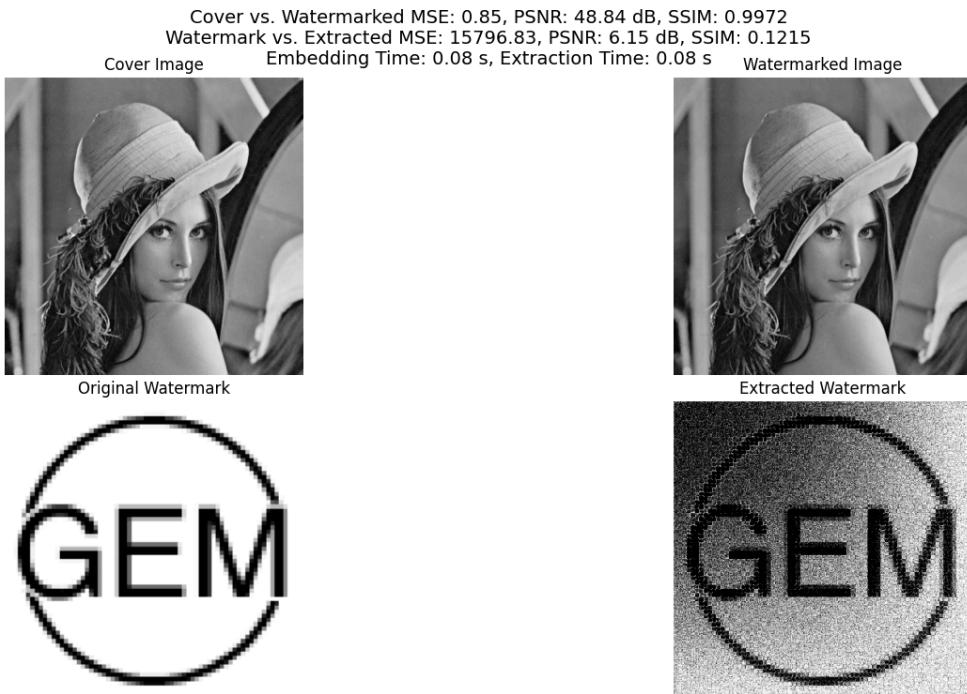
**Extraction Time: 0.43 secondi**

I tempi sono significativamente più lunghi rispetto al metodo LSB, a causa delle trasformazioni DCT e inverse su blocchi multipli. Il costo computazionale è quindi più elevato.

L'algoritmo DCT garantisce un'eccellente capacità di recupero del watermark anche in presenza di una maggiore alterazione dell'immagine host. Tuttavia, ciò comporta un compromesso in termini di qualità visiva (MSE e PSNR peggiori) e tempi di calcolo più lunghi. Il metodo risulta quindi più adatto a scenari in cui la priorità è la robustezza e non l'impercettibilità, come la protezione dei contenuti da modifiche o attacchi.

## DCT - Full

L'algoritmo applica la DCT 2D all'intera immagine, modifica tutti i coefficienti DCT sommando una versione scalata del watermark e infine applica la IDCT per ottenere l'immagine finale con watermark.



### Cover VS. Watermarked

#### ★ **MSE = 0.85**

Un MSE con risultato 0.85 è indice di una differenza media quadratica molto bassa tra i pixel, segno che il watermark ha introdotto modifiche lievi e ben distribuite.

#### ★ **PSNR = 48.08 dB**

Un PSNR di 48.08 indica una qualità molto alta. valore molto elevato che conferma un'alterazione impercettibile all'occhio umano (generalmente, valori >40 dB indicano immagini quasi indistinguibili).

#### ★ **SSIM = 0.9977**

Un SSIM con risultato 0.9977 risulta con una struttura quasi invariata che riflette una quasi totale conservazione della struttura visiva dell'immagine, comprensiva di luminanza, contrasto e texture locali.

## Watermark VS Extracted

### ★ MSE = 15796.54

Un valore di MSE pari a 15.797,54 indica un'estrazione imperfetta del watermark. Questo valore molto elevato evidenzia una significativa perdita di fedeltà nei pixel dell'immagine recuperata, segnalando un forte degrado rispetto all'originale.

### ★ PSNR = 6.15 dB

Il valore del PSNR è di 6,15 dB, un livello estremamente basso. Questo dato conferma una qualità molto scarsa dell'immagine ricostruita e suggerisce che il watermark è stato pesantemente compromesso durante il processo inverso.

### ★ SSIM = 0.1215

L'SSIM pari a 0,1215 indica una bassa similarità strutturale tra l'immagine originale e quella ottenuta. Tale valore, molto distante dall'ideale 1, suggerisce che la struttura originale del watermark è soltanto parzialmente riconoscibile.

L'immagine watermarked appare visivamente identica all'originale, mentre il watermark estratto è visibile ma degradato (rumore, distorsione nei dettagli fini).

L'algoritmo funziona particolarmente bene con immagini come Lena perché è ben bilanciata in termini di frequenze (né troppo piatta né troppo complessa), ha contrasti graduali e texture naturali, che assorbono bene le variazioni dei coefficienti DCT e non contiene grandi aree piatte, che altrimenti evidenzierebbero facilmente le alterazioni.

Un watermark semplice (come il logo "GEM") ha meno frequenze alte quindi è meno soggetto a perdita nel passaggio DCT-IDCT, è più robusto alla compressione ed è più facile da riconoscere visivamente anche se degradato.

Il buon risultato è dovuto a diversi fattori che hanno lavorato insieme in modo efficace.

L'uso della DCT su tutta l'immagine ha permesso di distribuire il watermark in modo uniforme, rendendolo meno visibile localmente e quindi meno percepibile all'occhio umano.

La scalatura del watermark, ovvero la riduzione della sua intensità prima dell'inserimento, ha aiutato a controllare la distorsione introdotta. Questo ha mantenuto alta la qualità visiva dell'immagine host.

Infine, c'è stata una buona combinazione tra l'immagine host e il watermark. Un'immagine bilanciata e un watermark semplice hanno reso più efficace il lavoro dell'algoritmo, migliorando il risultato complessivo.

La qualità del watermark estratto risente delle limitazioni intrinseche della trasformata: le informazioni ad alta frequenza (dettagli fini) sono le prime a degradarsi.

Per migliorare l'estrazione visiva si potrebbero usare delle tecniche di normalizzazione post-estrazione, filtraggio adattivo o modelli di enhancement basati su AI.

## DCT - JPEG - like

L'estrazione del DCT JPEG-like avviene leggendo i coefficienti post-quantizzazione, replicando il flusso della compressione JPEG. I bit del messaggio vengono recuperati dal LSB di coefficienti scelti che sopravvivono alla quantizzazione.



In questo approccio, l'algoritmo replica il comportamento della compressione JPEG: applica la DCT su blocchi  $8 \times 8$  dell'immagine, quantizza i coefficienti e inserisce i bit del messaggio segreto nei LSB dei coefficienti sopravvissuti alla quantizzazione. In fase di estrazione, l'informazione viene recuperata leggendo nuovamente i LSB dei coefficienti selezionati.

### ★ MSE = 25.69

Un MSE con valore 25.69 indica una modifica più consistente rispetto ad altri metodi, dovuta al fatto che il watermark viene inserito dopo la quantizzazione, su coefficienti più instabili o soggetti a variazione. Le differenze tra pixel originali e modificati risultano quindi più marcate, specialmente nei blocchi compressi in modo più aggressivo.

## ★ **PSNR = 34.00 dB**

Un valore di PSNR con 34,00 è ancora accettabile ma inferiore a quello degli altri metodi, riflettendo una perdita visiva più evidente, seppur contenuta.

In particolare, si possono notare leggere alterazioni in aree piatte o uniformi, dove il cambiamento dei coefficienti DCT è più percepibile.

## ★ **SSIM = 0.8677**

L'SSM a 0.8677 mostra che la struttura dell'immagine è in parte alterata, soprattutto nei pattern locali e nei dettagli fini. Il valore, pur rimanendo sopra 0.85, suggerisce una leggera perdita di coerenza visiva, soprattutto se confrontato con i metodi che operano direttamente in dominio spaziale o su tutta la DCT prima della quantizzazione.

## ★ **Accuratezza del Recupero**

Il logo "GEM" viene estratto perfettamente, come dimostrano i seguenti valori:

**MSE = 0.00**

**PSNR =  $\infty$**

**SSIM = 1.0000**

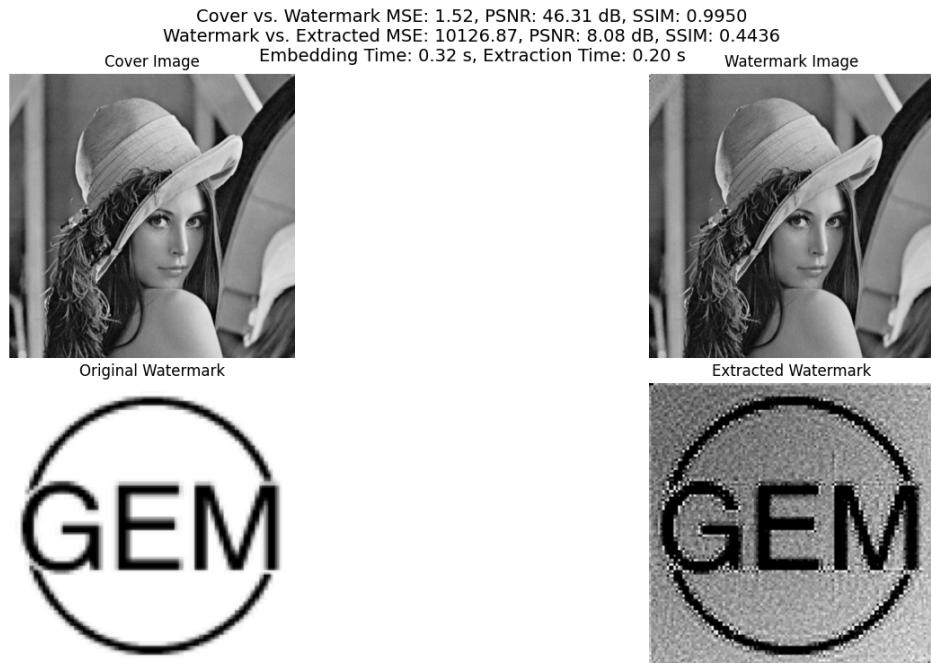
Ciò dimostra l'eccellente precisione del metodo di recupero, che riesce a leggere correttamente tutti i bit nascosti, grazie alla selezione strategica dei coefficienti meno soggetti a variazioni nella compressione.

Il metodo DCT JPEG-like è pensato per simulare un ambiente realistico di compressione, ed è quindi utile per testare la robustezza del watermark in scenari pratici. Tuttavia, comporta una perdita visiva maggiore rispetto ad altri approcci, rendendolo più adatto a casi in cui la precisione del recupero è prioritaria.

L'algoritmo funziona in modo ottimale quando le dimensioni dell'immagine sono potenze di 2, poiché la trasformata DCT e la suddivisione in blocchi  $8 \times 8$  avvengono in modo regolare, senza bisogno di padding o riadattamenti. Utilizzare dimensioni non allineate può ridurre la precisione o introdurre artefatti ai margini.

## DCT - Low frequency

L'estrazione del DCT-lowFrequency si concentra sui coefficienti a bassa frequenza, che sono più stabili e meno soggetti a perdita durante compressione o filtraggio. I bit vengono recuperati da questi coefficienti tramite lettura mirata.



In questo esperimento, il watermark è stato inserito selettivamente nei coefficienti a bassa frequenza della trasformata DCT, cioè nei componenti più stabili e meno suscettibili alla perdita in seguito a compressioni JPEG o trasformazioni aggressive.

Questa tecnica mira a migliorare la robustezza del watermark, garantendone la persistenza anche in caso di ri-salvataggi o manipolazioni leggere dell'immagine.

### Cover VS. Watermarked

#### ★ MSE = 1.52

Un MSE con valore 1.52 risulta con una modifica moderata, inferiore rispetto al metodo JPEG-like. Indica che l'alterazione dei coefficienti DCT è distribuita con maggiore attenzione, preservando la qualità generale dell'immagine.

#### ★ PSNR = 46.31 dB

Un PSNR di 46.31 è un valore molto elevato, che testimonia una qualità visiva eccellente. Le modifiche sono impercettibili a occhio nudo, soprattutto perché limitate alle frequenze meno visibili.

#### ★ SSIM = 0.9950

Il valore SSIM di 0.9950 indica un'ottima conservazione della struttura visiva e delle relazioni spaziali tra i pixel. L'immagine modificata risulta praticamente identica all'originale per un osservatore umano.

### Watermark VS Extracted

#### ★ MSE = 10126.87

L'MSE risultante 10126.87 tra l'immagine watermark originale e quella estratta è molto elevato. Questo significa che, pixel per pixel, ci sono grandi differenze numeriche, dovute in larga parte all'introduzione di rumore e distorsione durante il processo di embedding e successiva estrazione.

Valori così alti sono tipici nei casi in cui il watermark è strutturalmente complesso, come nel caso di immagini con linee nette e bordi definiti: anche piccole variazioni nei coefficienti possono generare grandi discrepanze in fase di ricostruzione.

#### ★ PSNR = 8.08 dB

Il PSNR con 8.08 risulta molto basso, il che indica che la quantità di disturbo rispetto al segnale originale è alta. A livello visivo, questo si traduce in un'immagine del watermark molto rumorosa, che perde i dettagli fini e i contrasti puliti.

Tuttavia, un PSNR basso non implica necessariamente l'inutilizzabilità: nel caso specifico, il messaggio rimane leggibile grazie alla solidità delle forme principali (es. cerchio e lettere), anche se la qualità complessiva è compromessa.

## ★ SSIM = 0.4436

L'SSIM di 0.4436 è inferiore a 0.5, indicando che la struttura locale e globale dell'immagine estratta differisce in modo significativo da quella originale. Le relazioni tra i pixel, il contrasto e la distribuzione delle forme non sono mantenute in modo fedele.

Il valore suggerisce che solo la forma generale del watermark è sopravvissuta, ma le caratteristiche visive fini sono state deformate. Questo riflette bene l'effetto di un embedding in bassa frequenza, che sacrifica i dettagli fini a favore della robustezza.

Il watermark è stato riconosciuto ma fortemente degradato, a causa del suo alto contenuto di frequenze (linee e bordi netti). Questo tipo di contenuto viene mal rappresentato quando inserito in coefficienti a bassa frequenza, che trasportano informazioni più "morbide" o graduali.

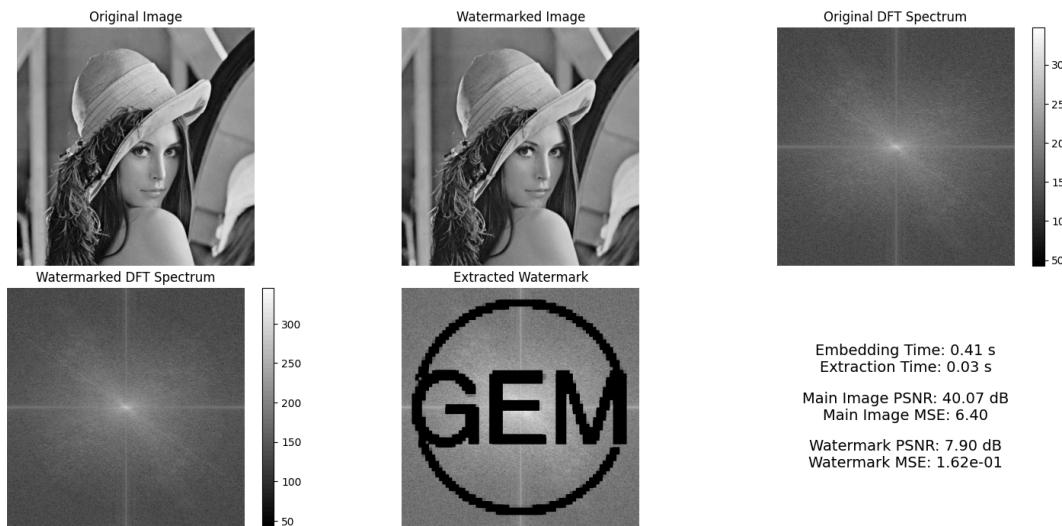
Un possibile miglioramento potrebbe essere sfocare lievemente il watermark prima dell'inserimento, riducendone il contenuto frequenziale e adattandolo meglio al dominio scelto. In questo modo si migliora la fedeltà dell'estrazione, mantenendo leggibilità e compatibilità con l'algoritmo.

L'approccio DCT\_lowFrequency rappresenta un ottimo compromesso tra robustezza e impercettibilità. È particolarmente adatto in scenari in cui l'immagine potrebbe essere compressa o modificata, ma dove è comunque importante mantenere l'integrità estetica dell'immagine host.

E' stato anche testato con un watermark più creativo: "Lena dentro Lena". Questo caso particolare dimostra come watermark visivamente simili alla cover image si nascondano meglio e risultino esteticamente coerenti, pur con un'efficacia d'estrazione limitata dalla complessità dell'immagine segreta.

## DFT

L'estrazione del DFT si basa sull'analisi della magnitudine dei coefficienti di Fourier, dove il messaggio è stato nascosto. La trasformata inversa non è necessaria per leggere i dati, che vengono ricavati confrontando variazioni nelle ampiezze. La robustezza alle trasformazioni geometriche globali rende questo metodo adatto a scenari dinamici.



In questo esperimento, il watermark è stato inserito nel dominio della trasformata discreta di Fourier (DFT). La tecnica sfrutta le caratteristiche della trasformata per modificare lo spettro dell'immagine, nascondendo l'informazione all'interno dei suoi coefficienti complessi.

Le immagini mostrano il confronto tra l'immagine originale e quella watermarked, i rispettivi spettri DFT e il watermark originale e quello estratto.

### ★ Main Image PSNR = 40.07 dB

Il valore di PSNR è pari a 40,07 dB, un risultato molto positivo. In generale, un PSNR superiore a 40 dB è considerato indicativo di una buona qualità visiva, poiché le alterazioni introdotte sono minime e difficilmente percepibili dall'occhio umano.

### ★ Main Image MSE = 6.40

Il valore di MSE, invece, è pari a 6,40, un dato molto contenuto. Questo valore conferma che la distorsione media per pixel è bassa e perfettamente in linea con le prestazioni attese da metodi non invasivi di watermarking.

### ★ Watermark PSNR = 7.90 dB

Il valore di PSNR del watermark è pari a 7,90 dB, un dato relativamente basso che indica una leggera degradazione visiva dell'immagine estratta.

### ★ Watermark MSE = 0.162

L'MSE è molto contenuto, con un valore di 0,162, il che suggerisce che i dati numerici sono comunque molto vicini all'originale.

Nonostante la qualità visiva non sia elevata, la forma del logo "GEM" risulta perfettamente riconoscibile, a conferma del fatto che il watermark è stato estratto con successo.

### ★ Tempi di elaborazione

**Embedding Time = 0.41 s**

**Extraction Time = 0.03 s**

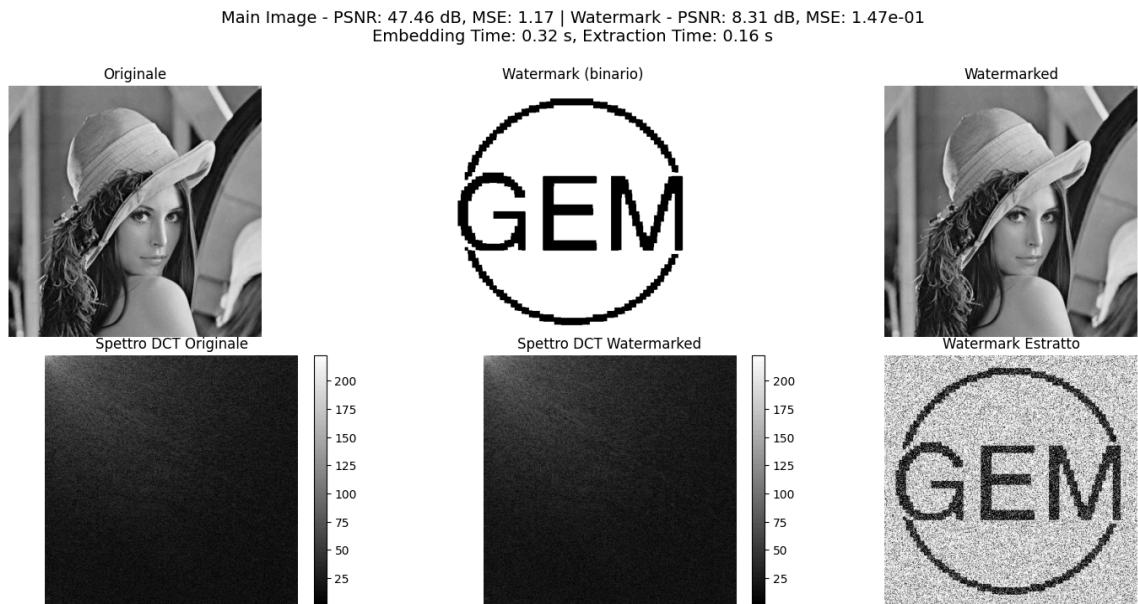
I tempi indicano un'implementazione relativamente efficiente, compatibile con applicazioni in tempo quasi reale o batch processing.

Nel complesso, non si rilevano criticità significative nell'utilizzo dell'algoritmo DFT. Il metodo si dimostra efficace e ben bilanciato, riuscendo a mantenere una buona qualità visiva dell'immagine host, pur garantendo una robusta e chiara estrazione del watermark.

Inoltre, l'intervento risulta ben rappresentato anche nello spettro frequenziale: le modifiche introdotte sono visibili nelle immagini DFT, segno di un'alterazione mirata e coerente nel dominio delle frequenze.

## DSSS

L'estrazione del DSSS avviene tramite correlazione tra la sequenza pseudo-casuale originale e i coefficienti modificati nel dominio DCT. Ogni bit del messaggio è stato diffuso su più punti, rendendo possibile il recupero anche in presenza di rumore o perdite locali. Questo garantisce una robustezza elevata e ridondanza del segnale.



In questo esperimento è stato utilizzato un metodo di watermarking basato su DSSS, una tecnica derivata dalle telecomunicazioni, che diffonde il messaggio segreto all'interno dell'immagine attraverso una sequenza pseudo-casuale (PN sequence). Questo approccio mira a garantire robustezza e segretezza, distribuendo l'informazione in modo sparso e poco prevedibile all'interno del dominio dell'immagine (qui trasformato con DCT).

### Main Image (Cover vs. Watermarked)

#### ★ PSNR = 47.46 dB

Il valore di PSNR con 47.46 è molto alto, il che indica che la qualità dell'immagine è eccellente. Con un PSNR>40 dB, le differenze tra l'immagine originale e quella con watermark sono impercettibili all'occhio umano.

## ★ **MSE = 1.17**

L'MSE risultante 1.17 è molto basso, segnalando che le modifiche introdotte nei pixel sono minime. L'inserimento del watermark non ha compromesso la fedeltà visiva della cover image.

L'immagine watermarked mantiene una qualità visiva praticamente indistinguibile dall'originale, grazie a un inserimento ben calibrato nei coefficienti meno sensibili.

### Watermark (Original vs. Estratto)

## ★ **PSNR = 8.31 dB**

Questo valore di PSNR di 8.31 è molto basso, segnalando che l'immagine estratta è pesantemente degradata rispetto al watermark originale. Tuttavia, un PSNR così basso non significa necessariamente che il contenuto non sia interpretabile: il messaggio può essere ancora visivamente riconoscibile.

## ★ **MSE = 0.147**

L'MSE di 0.147 risulta relativamente contenuto, indicando che le differenze numeriche tra i pixel non sono estreme. Questo può avvenire in immagini ad alto contrasto come un logo binario (nero su bianco), dove anche piccole variazioni introducono rumore visivo evidente.

Il watermark è stato recuperato con successo, ma risulta affetto da rumore distribuito, che ne degrada l'aspetto. Le informazioni principali (es. il testo "GEM") sono distinguibili, ma non fedeli all'originale. È un risultato tipico di metodi orientati alla robustezza più che alla qualità visiva.

I risultati ottenuti con il metodo DSSS possono apparire meno soddisfacenti rispetto ad altri approcci, ma questo è legato alla natura stessa della tecnica. L'obiettivo principale del DSSS non è ottenere un'estrazione visivamente perfetta, bensì garantire che il messaggio sia ben nascosto, difficile da rilevare e resistente a eventuali attacchi o manipolazioni.

In particolare, il messaggio viene distribuito su tutta l'immagine e successivamente ricostruito tramite correlazione. Questo processo, per sua stessa struttura, introduce un certo grado di rumore pseudo-casuale.

Di conseguenza, l'immagine del messaggio estratto risulta sì visibile, ma disturbata, con una qualità che riflette questo compromesso tra invisibilità e robustezza.

Per migliorare la qualità dell'estrazione del messaggio nel metodo DSSS, si potrebbero adottare alcune soluzioni mirate. Una prima possibilità è integrare un sistema di denoising nella fase di decodifica, utilizzando tecniche come filtri mediani, sogliatura adattiva o modelli neurali, che aiuterebbero a ridurre il rumore introdotto dalla correlazione e a rendere il messaggio più leggibile.

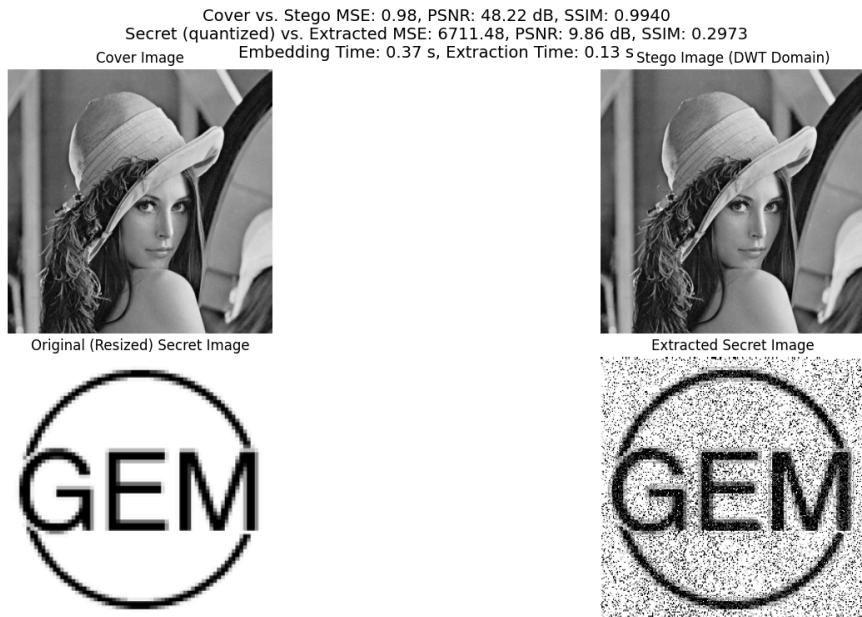
Un altro intervento utile riguarda l'ottimizzazione della sequenza pseudo-casuale (PN) e del rapporto segnale-rumore impiegato nella modulazione del watermark, con l'obiettivo di migliorare la qualità dell'estrazione senza compromettere la discrezione del messaggio.

Infine, si potrebbe aumentare la ridondanza del messaggio o applicare un embedding adattivo, limitandosi alle zone dell'immagine con maggiore varianza, dove le modifiche risultano meno percepibili all'occhio umano. Queste strategie contribuirebbero a un risultato visivamente più pulito e robusto.

Il metodo DSSS conferma la sua vocazione alla segretezza e alla resilienza, anche se a scapito della qualità visiva del watermark estratto. Nonostante il risultato sia visivamente più debole rispetto ad altre tecniche, il logo "GEM" risulta comunque leggibile, segno che il messaggio è stato correttamente codificato e ricostruito.

## DWT - OneLevel

L'estrazione del DWT one-level consiste nell'applicare nuovamente la trasformata wavelet per isolare la sottobanda cH, dove sono stati nascosti i bit del messaggio. I dati vengono recuperati leggendo i bit meno significativi dei coefficienti. La combinazione tra dominio spaziale e frequenziale offre una buona resistenza a modifiche localizzate.



Il watermark è stato inserito nel dominio della trasformata discreta wavelet (DWT), utilizzando un'unica decomposizione a livello 1. A differenza delle tecniche basate su DCT o DSSS, la DWT scomponete l'immagine in sotto-bande (approssimazione e dettagli) in modo gerarchico, permettendo di modificare l'informazione in frequenze spaziali differenti.

### Cover vs. Stego Image

★ **MSE = 0.98**

L'MSE di 0.98 è estremamente basso, a conferma del fatto che le modifiche introdotte sono minime. Questo valore vicino a 0 implica che, a livello numerico, i pixel sono stati modificati pochissimo.

### ★ PSNR = 48.22 dB

Un PSNR di 40.22 indica una rapporto segnale-rumore molto alto. Valori > 40 dB sono considerati eccellenti e indicano che le alterazioni sono praticamente impercettibili a livello visivo. Il watermark è nascosto in modo efficace e trasparente.

### ★ SSIM = 0.9940

L'SSIM risultante 0.9940 mostra una quasi totale corrispondenza tra le strutture visive delle due immagini. Variazioni locali di luminanza, contrasto e texture sono quasi inesistenti.

L'algoritmo DWT a un livello si conferma altamente invisibile, distribuendo le modifiche in regioni frequenziali poco sensibili. L'occhio umano non percepisce differenze, e la qualità visiva della cover è mantenuta.

### Secret (quantized) vs. Extracted Image

### ★ MSE = 6711.48

L'MSE di 6711.48 segnala un valore molto alto: i pixel dell'immagine estratta sono molto diversi rispetto a quelli del watermark originale. Questo si traduce in un'elevata presenza di rumore e distorsione nella ricostruzione.

### ★ PSNR = 9.86 dB

Il PSNR di 9.86, invece, segnala un valore molto basso: indica che il segnale del watermark recuperato è molto contaminato dal rumore. A livello visivo, il logo è ancora distinguibile, ma non più nitido o fedele all'originale.

### ★ SSIM = 0.2973

L'SSIM di 0.2973 è molto basso, indicando una perdita significativa della coerenza visiva tra watermark originale ed estratto. Solo la forma globale è stata conservata.

L'immagine segreta è stata recuperata solo parzialmente. Il messaggio è leggibile, ma compromesso da forte rumore diffuso. L'informazione è sopravvissuta solo a livello strutturale, perdendo però precisione e pulizia nei dettagli.

## DWT - MultiLevel

L'estrazione del DWT multi-level prevede una scomposizione ricorsiva dell'immagine per ottenere tutte le sottobande di dettaglio. I bit del messaggio vengono recuperati leggendo i LSB dei coefficienti distribuiti su più livelli. La struttura multi-scala migliora la capacità di embedding e la robustezza all'elaborazione dell'immagine.



In questo test, il watermark è stato inserito nel dominio DWT utilizzando una wavelet biortogonale e una decomposizione a più livelli. L'obiettivo di questo approccio è sfruttare le componenti direzionali (approssimazione, dettaglio orizzontale, verticale e diagonale) per nascondere l'informazione in zone meno sensibili dell'immagine.

### Cover vs. Stego Image

#### ★ **MSE = 4.31**

L'MSE di 4.31 risulta contenuto, segnalando una modifica visibile ma non eccessiva nei pixel dell'immagine.

#### ★ **PSNR = 41.79 dB**

Un valore PSNR uguale a 41.79 è appena sopra la soglia di 40 dB, indica una buona qualità visiva, anche se inferiore rispetto a embedding DWT più semplici.

## ★ SSIM = 0.9747

L'SSIM di 0.9747 indica che l'immagine mantiene gran parte della struttura originale, con leggere variazioni percepibili nei pattern locali. Buon compromesso tra qualità e capacità d'inserimento.

### Secret vs. Extracted Image

## ★ MSE = 3676.01

Il valore di MSE è pari a 3676,01, il che indica che il watermark è stato recuperato con un livello di rumore moderato, ma senza una perdita completa dell'informazione.

## ★ PSNR = 12.48 dB

Il PSNR è di 12,48 dB, un valore che segnala una degradazione del segnale, ma comunque tale da rendere il watermark ancora visivamente interpretabile.

## ★ SSIM = 0.2971

L'SSIM è pari a 0,2971, un valore piuttosto basso che conferma che la struttura fine del watermark, come spigoli e bordi netti, è stata alterata. Tuttavia, le componenti principali dell'immagine risultano ancora presenti e riconoscibili.

L'immagine utilizzata come watermark, contenente la scritta "GEM", presenta molte linee orizzontali e curve ben definite. Questi elementi tendono a sollecitare in modo marcato i coefficienti orizzontali e diagonali all'interno dei sottoblocchi ottenuti tramite la trasformata wavelet.

Le bande visibili che si notano nell'immagine stego derivano proprio dalla modifica diretta dei coefficienti DWT, in particolare di quelli associati ai dettagli nei livelli superiori della decomposizione. Queste alterazioni, pur necessarie per l'inserimento del messaggio, possono introdurre artefatti percettibili a livello visivo.

Un aspetto fondamentale è la scelta della combinazione tra numero di livelli e tipo di wavelet. Dalle osservazioni sperimentali, è emerso che le configurazioni più efficaci sono quelle che utilizzano 2 o 3 livelli di decomposizione insieme alla wavelet. Questa combinazione riesce a bilanciare bene la capacità di inserimento del watermark con un contenimento degli artefatti visivi, offrendo così un buon compromesso tra qualità e robustezza.

L'impiego della trasformata DWT su più livelli offre una maggiore flessibilità nell'inserimento del watermark, permettendo di agire su diverse scale spaziali dell'immagine. Questa caratteristica consente un controllo più preciso tra visibilità e robustezza del segnale nascosto.

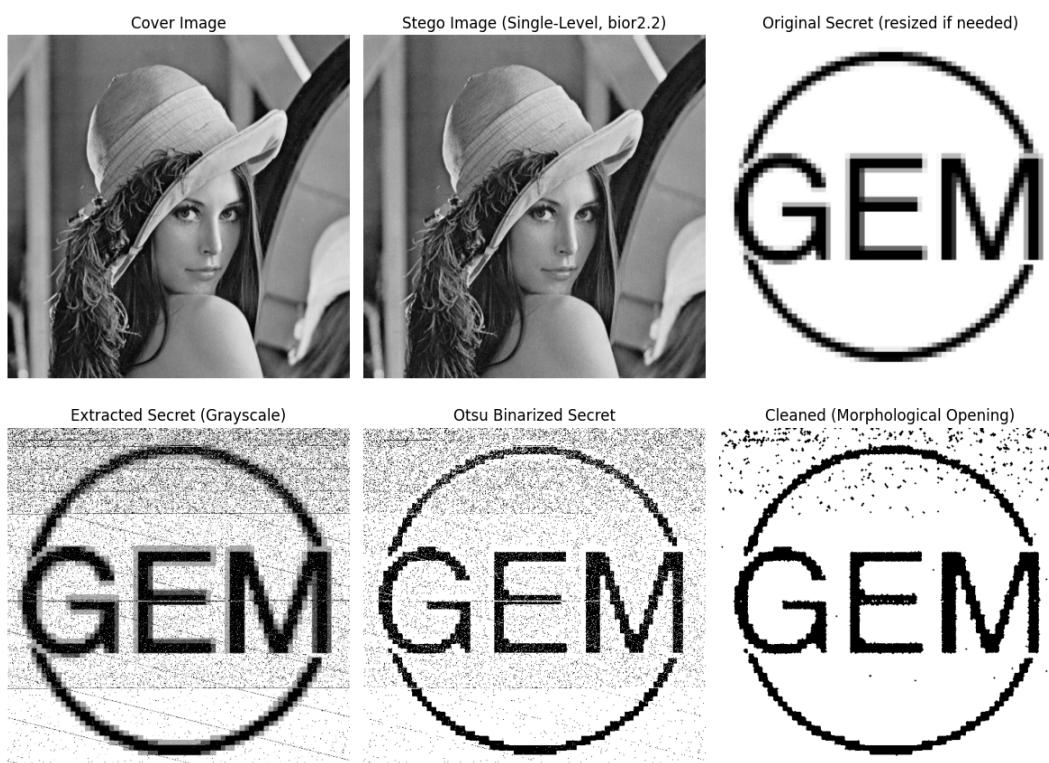
Tuttavia, l'approccio non è privo di effetti collaterali: in presenza di watermark con geometrie ben definite, come il logo "GEM", possono emergere pattern visibili, sotto forma di bande o disturbi localizzati. Questi artefatti sono direttamente legati sia alla scelta dei coefficienti modificati, sia alla forma del watermark stesso.

Nel caso specifico, la leggibilità del watermark rimane buona e il contenuto è chiaramente riconoscibile, ma si nota una certa alterazione nella qualità visiva complessiva, dovuta a un rumore non sempre uniforme. Questo evidenzia l'importanza di un bilanciamento attento tra inserimento efficace e mantenimento dell'estetica dell'immagine host.

## DWT - OTSU

L'estrazione del DWT-otsu utilizza la decomposizione multi-livello per recuperare il watermark dalle sottobande di dettaglio, seguita dall'applicazione della soglia di Otsu per binarizzare l'immagine estratta. Successive operazioni morfologiche eliminano il rumore residuo. Questo garantisce una ricostruzione nitida e precisa di watermark binari.

Cover vs. Stego: MSE=4.34, PSNR=41.75 dB, SSIM=0.9746  
Secret vs. Extracted: MSE=2990.72, PSNR=13.37 dB, SSIM=0.6552  
Embedding Time: 1.09 s, Extraction Time: 0.51 s



In questo esperimento, il watermark viene inserito nel dominio multilivello DWT, sfruttando diverse bande di frequenza, e successivamente viene binarizzato tramite soglia automatica di Otsu, seguita da una pulizia morfologica per migliorare la leggibilità.

### Cover vs. Stego Image

#### ★ MSE = 4.34

Il valore di MSE è pari a 4,34, un risultato contenuto che rientra nei valori tipici ottenuti con metodi basati sulla trasformata DWT. Questo indica che le modifiche introdotte sono ben distribuite e poco percettibili all'occhio umano.

#### ★ PSNR = 41.75 dB

Il PSNR è di 41,75 dB, un valore superiore alla soglia di 40 dB, comunemente considerata indicativa di eccellente qualità visiva. Ciò conferma che l'immagine stego mantiene un aspetto naturale e non mostra alterazioni evidenti.

#### ★ SSIM = 0.9746

Il valore di SSIM è 0,9746, segnalando che la struttura visiva dell'immagine di copertura è stata ben conservata, senza variazioni rilevanti nella texture o nei contorni.

### Secret vs. Extracted Image

#### ★ MSE = 2599.94

Il valore di MSE è pari a 2599,94 è piuttosto alto, segnalando la presenza di rumore nell'immagine estratta. Questo conferma che l'immagine in scala di grigi ottenuta inizialmente è degradata e non perfettamente fedele all'originale.

#### ★ PSNR = 13.37 dB

Un valore di PSNR di 13.37 è relativamente basso, che indica una qualità numerica non eccellente. Tuttavia, l'immagine conserva ancora un certo grado di visibilità e interpretabilità.

#### ★ SSIM = 0.6552

Questo valore di PSNR di 0.6552 indica una similarità strutturale moderata. La struttura visiva del watermark non è del tutto mantenuta, ma le caratteristiche principali restano riconoscibili.

Nonostante questi valori, l'applicazione della binarizzazione automatica e dell'opening morfologico ha permesso di ripulire l'immagine e di rendere il logo "GEM" perfettamente leggibile. La percezione visiva migliora nettamente nella versione elaborata, compensando le carenze nei valori numerici.

Il metodo si dimostra efficace nel limitare artefatti visivi evidenti: grazie alla distribuzione del watermark attraverso la trasformata DWT, non si notano bande regolari o disturbi marcati nell'immagine.

L'effetto della sogliatura automatica con il metodo di Otsu è presente e fornisce risultati accettabili, anche se non elimina completamente il rumore generato durante l'estrazione.

Infine, l'applicazione dell'opening morfologico migliora in modo evidente la leggibilità del watermark, ma comporta anche la perdita di alcune informazioni deboli, specialmente nei bordi del messaggio.

Il metodo basato su DWT multilevel combinato con la sogliatura di Otsu offre risultati tutto sommato nella media, senza particolari punti di forza rispetto ad altri approcci analizzati.

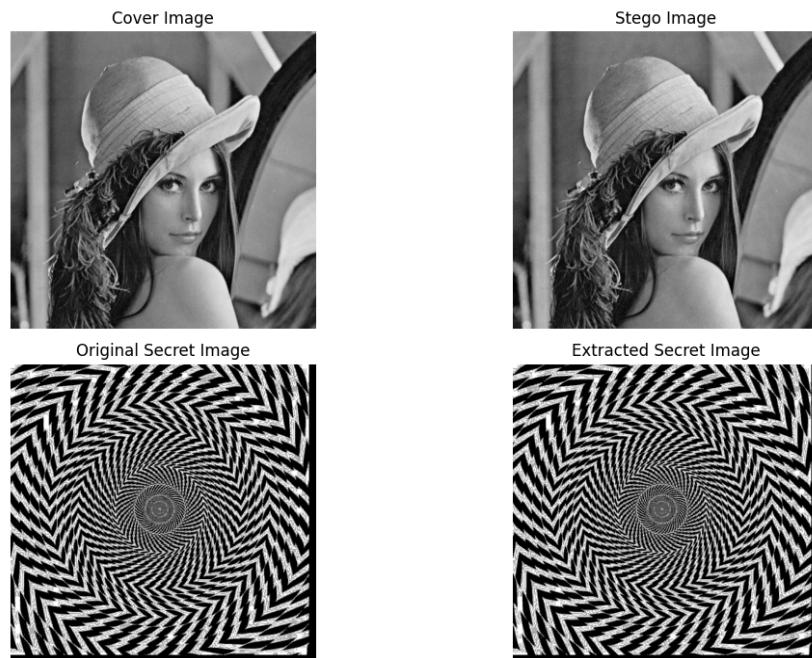
La qualità visiva dell'immagine host resta buona, e il watermark estratto risulta leggibile, anche se la resa non appare particolarmente raffinata o robusta in assenza di post-processing.

Nel complesso, si tratta di una soluzione di base, adatta a contesti poco critici, ma con margini di miglioramento. Risultati più solidi potrebbero essere ottenuti adottando, ad esempio, binarizzazione adattiva locale o tecniche di ricostruzione più avanzate.

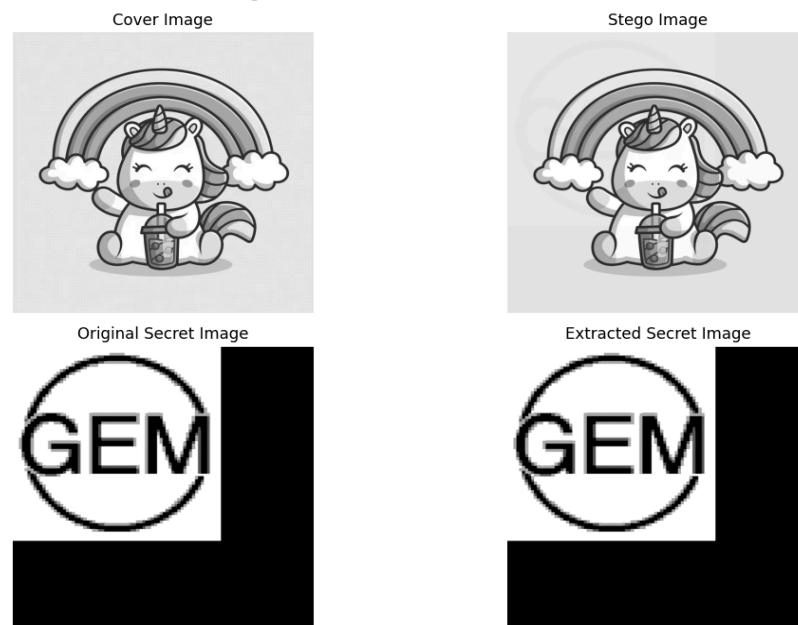
# Capitolo 4 - Risultati a confronto

## LSB - Spatial

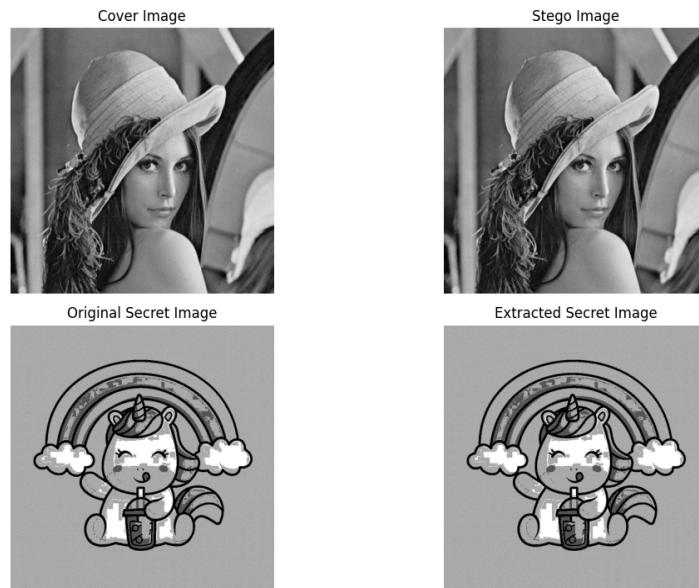
Original vs. Stego Image MSE: 2.74, PSNR: 43.75 dB, SSIM: 0.9862  
Original vs. Extracted Secret MSE: 0.00, PSNR: inf dB, SSIM: 1.0000  
Embedding Time: 0.00354s, Extraction Time: 0.00278s



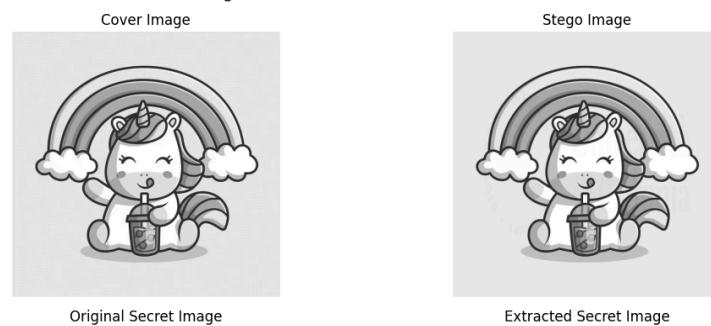
Original vs. Stego Image MSE: 3.07, PSNR: 43.26 dB, SSIM: 0.9980  
Original vs. Extracted Secret MSE: 0.00, PSNR: inf dB, SSIM: 1.0000  
Embedding Time: 0.00761s, Extraction Time: 0.00540s



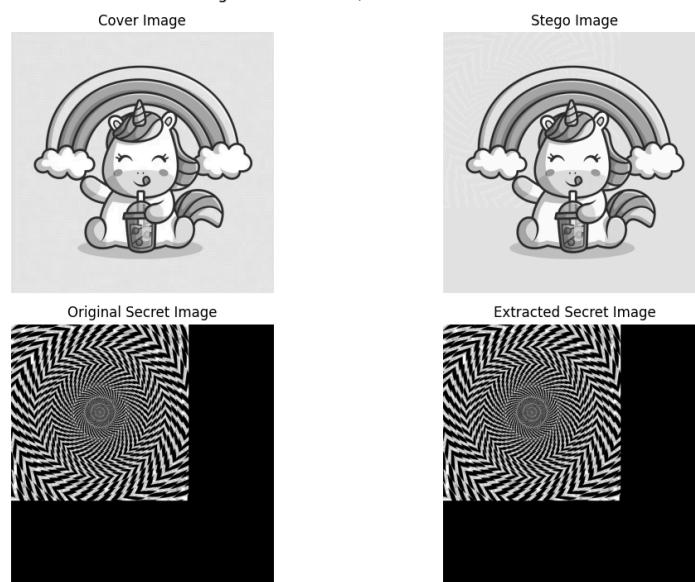
Original vs. Stego Image MSE: 1.76, PSNR: 45.67 dB, SSIM: 0.9913  
Original vs. Extracted Secret MSE: 0.00, PSNR: inf dB, SSIM: 1.0000  
Embedding Time: 0.00362s, Extraction Time: 0.00246s



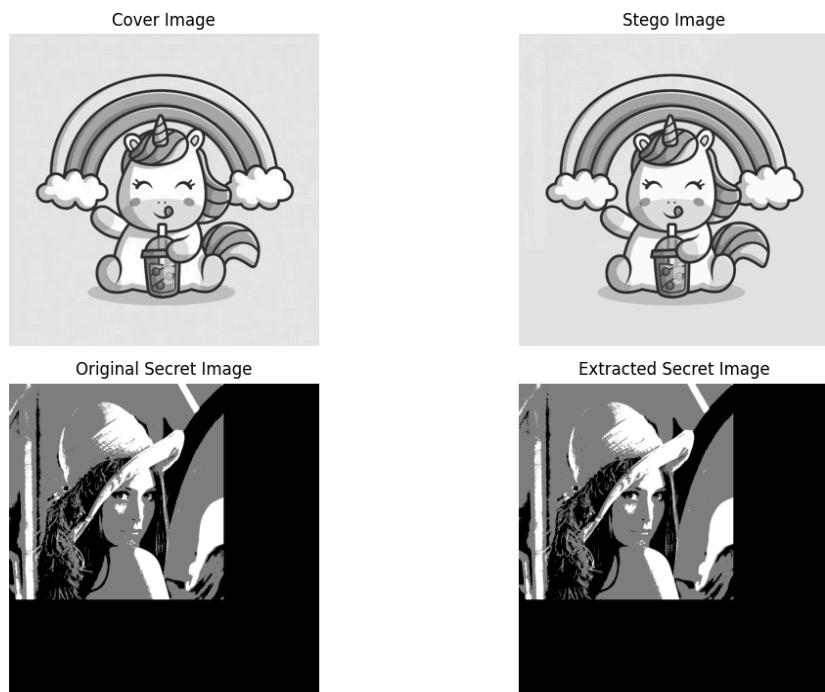
Original vs. Stego Image MSE: 1.79, PSNR: 45.61 dB, SSIM: 0.9975  
Original vs. Extracted Secret MSE: 0.00, PSNR: inf dB, SSIM: 1.0000  
Embedding Time: 0.00708s, Extraction Time: 0.00560s



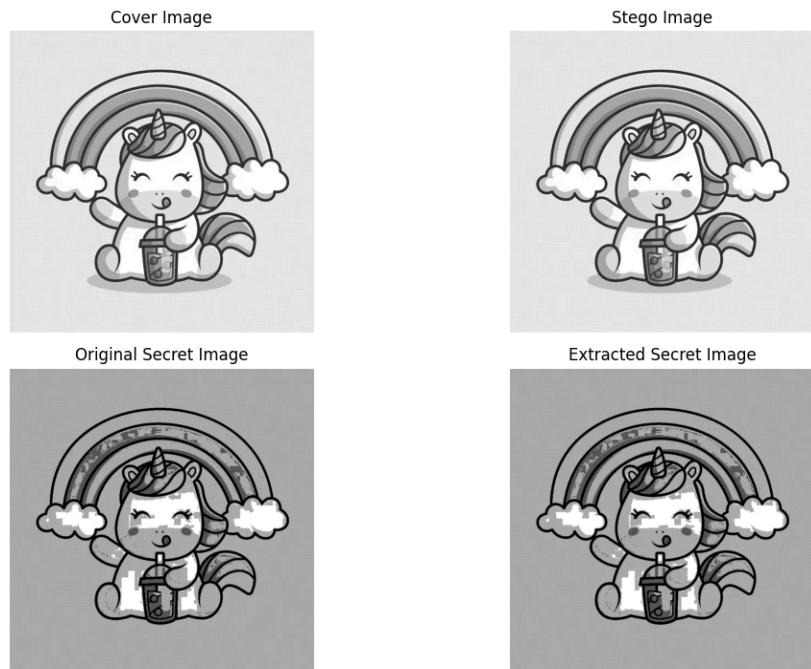
Original vs. Stego Image MSE: 3.36, PSNR: 42.87 dB, SSIM: 0.9942  
Original vs. Extracted Secret MSE: 0.00, PSNR: inf dB, SSIM: 1.0000  
Embedding Time: 0.00770s, Extraction Time: 0.00607s



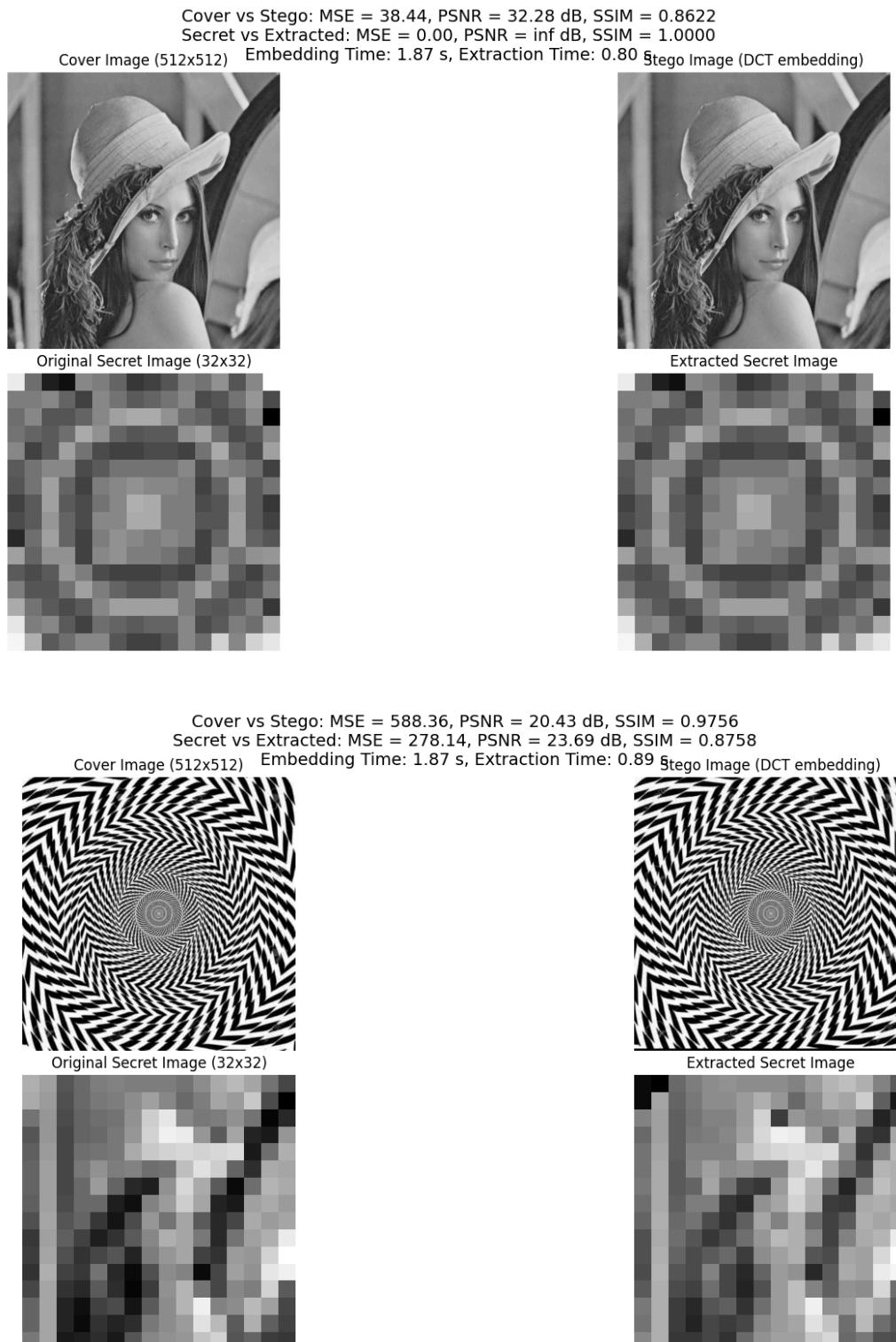
Original vs. Stego Image MSE: 3.06, PSNR: 43.27 dB, SSIM: 0.9986  
Original vs. Extracted Secret MSE: 0.00, PSNR: inf dB, SSIM: 1.0000  
Embedding Time: 0.00766s, Extraction Time: 0.00548s



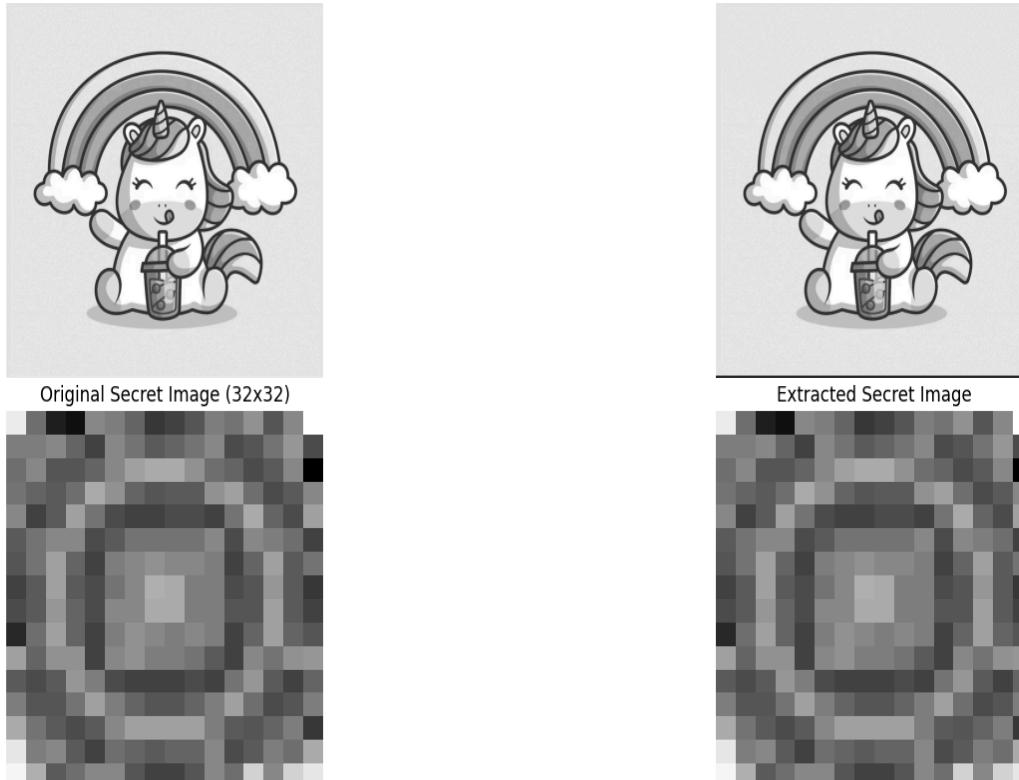
Original vs. Stego Image MSE: 0.57, PSNR: 50.59 dB, SSIM: 0.9991  
Original vs. Extracted Secret MSE: 0.00, PSNR: inf dB, SSIM: 1.0000  
Embedding Time: 0.00711s, Extraction Time: 0.00535s



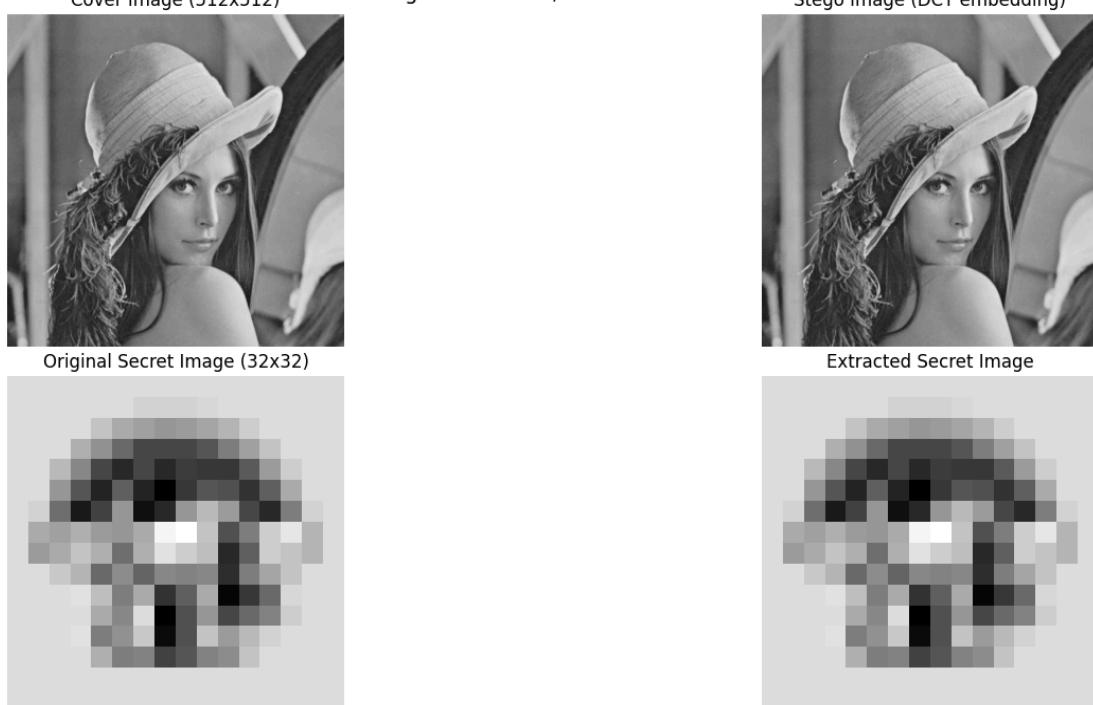
## DCT - Single Quant



Cover vs Stego: MSE = 601.53, PSNR = 20.34 dB, SSIM = 0.8770  
Secret vs Extracted: MSE = 0.00, PSNR = inf dB, SSIM = 1.0000  
Cover Image (512x512) Embedding Time: 3.89 s, Extraction Time: 2.27 s Stego Image (DCT embedding)



Cover vs Stego: MSE = 45.35, PSNR = 31.56 dB, SSIM = 0.8443  
Secret vs Extracted: MSE = 0.00, PSNR = inf dB, SSIM = 1.0000  
Cover Image (512x512) Embedding Time: 1.78 s, Extraction Time: 0.85 s Stego Image (DCT embedding)



Cover vs Stego: MSE = 65.11, PSNR = 29.99 dB, SSIM = 0.8001

Secret vs Extracted: MSE = 0.00, PSNR = inf dB, SSIM = 1.0000

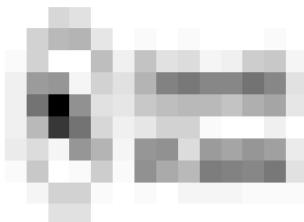
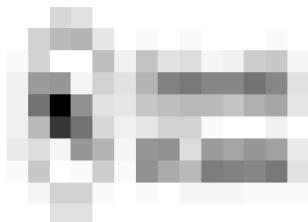
Cover Image (512x512) Embedding Time: 1.80 s, Extraction Time: 1.22 s Stego Image (DCT embedding)



Original Secret Image (32x32)



Extracted Secret Image



## DCT - Full

Cover vs. Watermarked MSE: 0.80, PSNR: 49.08 dB, SSIM: 0.9977  
Watermark vs. Extracted MSE: 14776.69, PSNR: 6.44 dB, SSIM: 0.1123

Cover Image

Embedding Time: 0.07 s, Extraction Time: 0.08 s

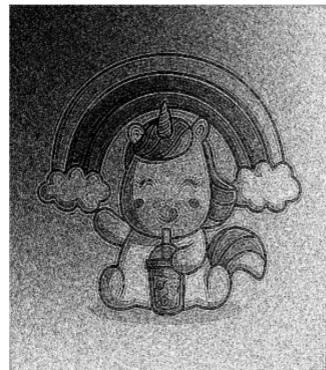
Watermarked Image



Original Watermark



Extracted Watermark



Cover vs. Watermarked MSE: 0.75, PSNR: 49.38 dB, SSIM: 0.9970  
Watermark vs. Extracted MSE: 5332.90, PSNR: 10.86 dB, SSIM: 0.6871

Cover Image

Embedding Time: 0.07 s, Extraction Time: 0.08 s

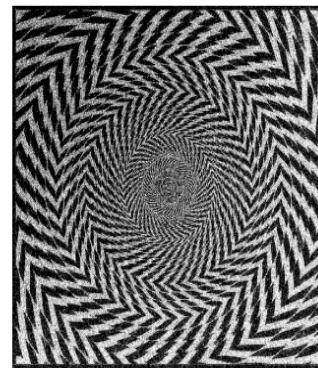
Watermarked Image

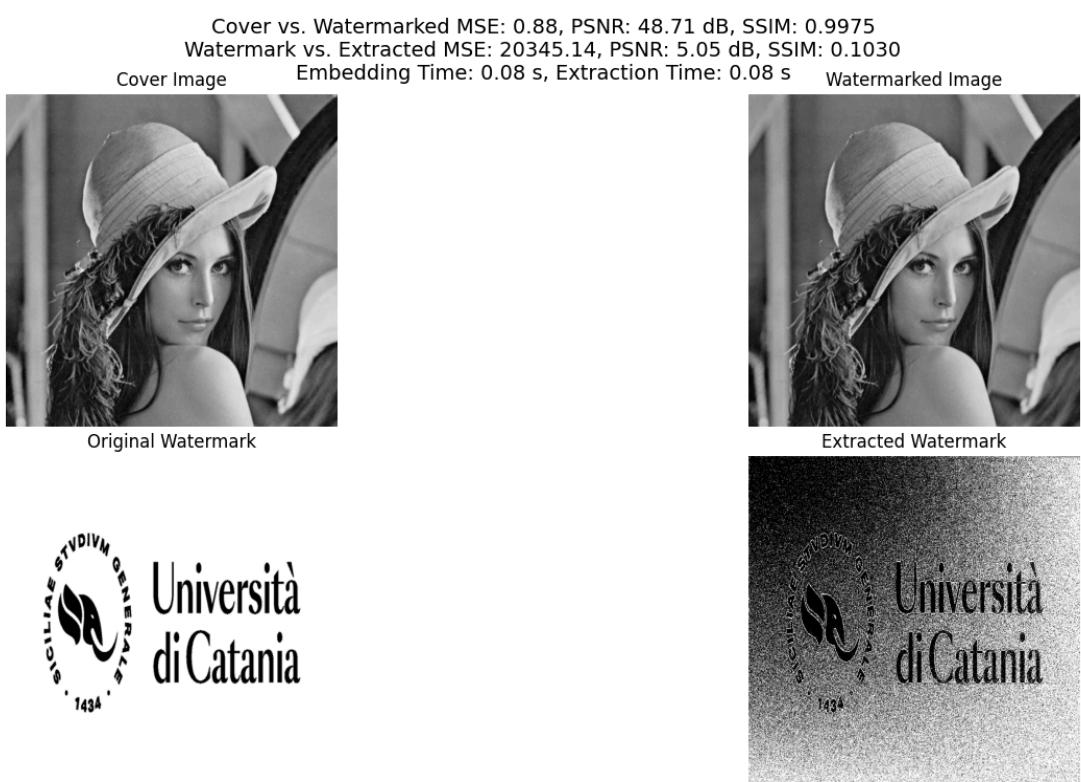


Original Watermark



Extracted Watermark





Cover vs. Watermarked MSE: 0.64, PSNR: 50.08 dB, SSIM: 0.9998  
 Watermark vs. Extracted MSE: 31020.76, PSNR: 3.21 dB, SSIM: 0.0761  
 Cover Image      Embedding Time: 0.07 s, Extraction Time: 0.06 s      Watermarked Image



Università  
di Catania



Cover vs. Watermarked MSE: 0.49, PSNR: 51.24 dB, SSIM: 0.9998  
 Watermark vs. Extracted MSE: 9650.45, PSNR: 8.29 dB, SSIM: 0.0985  
 Cover Image      Embedding Time: 0.06 s, Extraction Time: 0.07 s      Watermarked Image



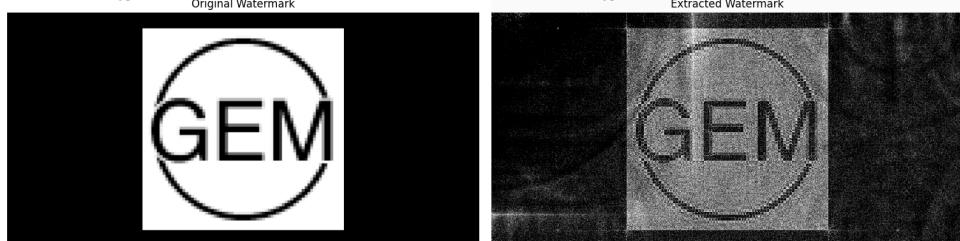
Cover vs. Watermarked MSE: 0.61, PSNR: 50.25 dB, SSIM: 0.9960  
 Watermark vs. Extracted MSE: 8076.26, PSNR: 9.06 dB, SSIM: 0.0909  
 Cover Image      Embedding Time: 0.22 s, Extraction Time: 0.24 s      Watermarked Image



Università  
di Catania



Università  
di Catania



## DCT - Jpeg - Like

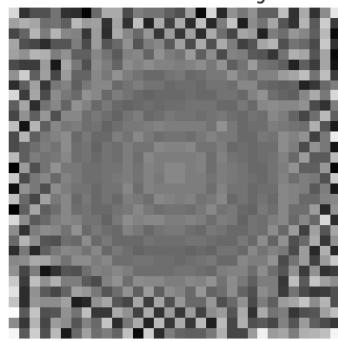
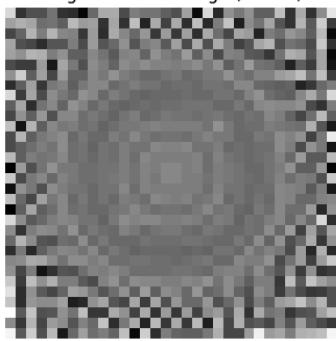
Cover vs Stego: MSE = 15.92, PSNR = 36.11 dB, SSIM = 0.9165  
Secret vs Extracted: MSE = 0.00, PSNR = inf dB, SSIM = 1.0000  
Cover Image (512x512) Embedding Time: 1.91 s, Extraction Time: 0.84 Stego Image (DCT embedding)



Original Secret Image (32x32)



Extracted Secret Image



Cover vs Stego: MSE = 595.20, PSNR = 20.38 dB, SSIM = 0.8762  
Secret vs Extracted: MSE = 171.64, PSNR = 25.78 dB, SSIM = 0.9462  
Cover Image (512x512) Embedding Time: 3.54 s, Extraction Time: 1.77 Stego Image (DCT embedding)



Original Secret Image (32x32)



Extracted Secret Image



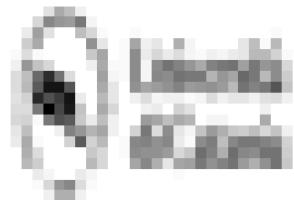
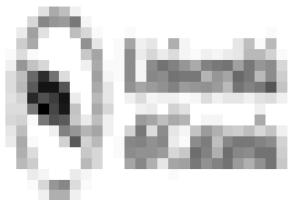
Cover vs Stego: MSE = 27.93, PSNR = 33.67 dB, SSIM = 0.8557  
Secret vs Extracted: MSE = 0.00, PSNR = inf dB, SSIM = 1.0000  
Cover Image (512x512) Embedding Time: 1.91 s, Extraction Time: 1.20 s Stego Image (DCT embedding)



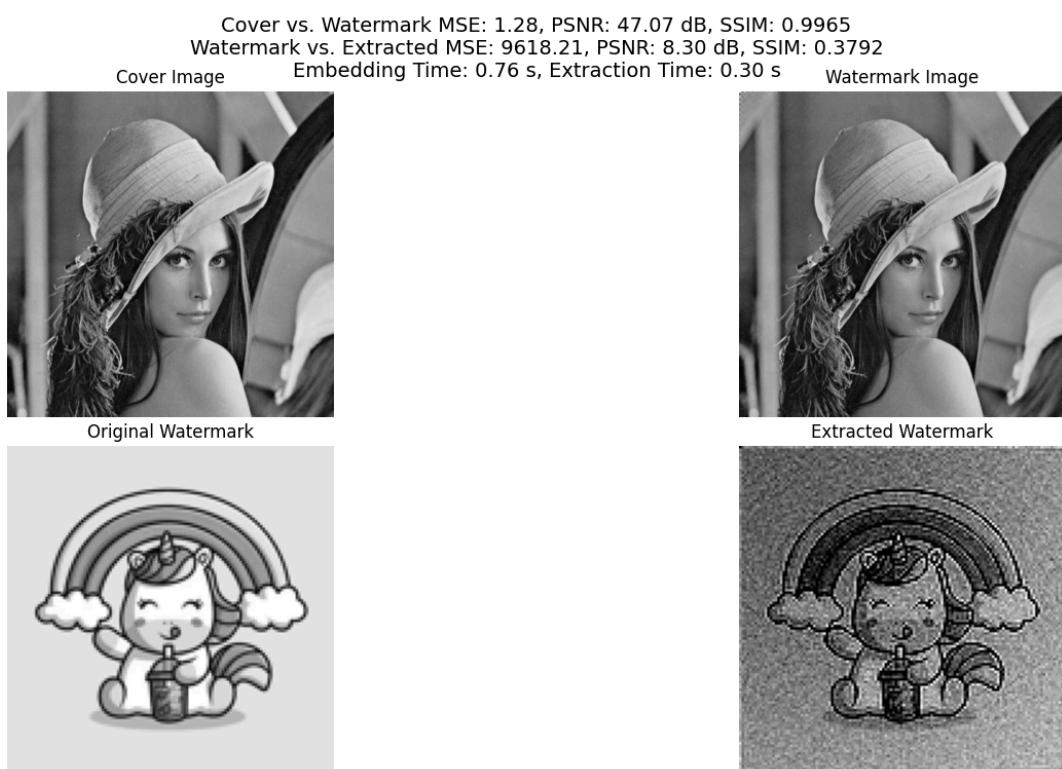
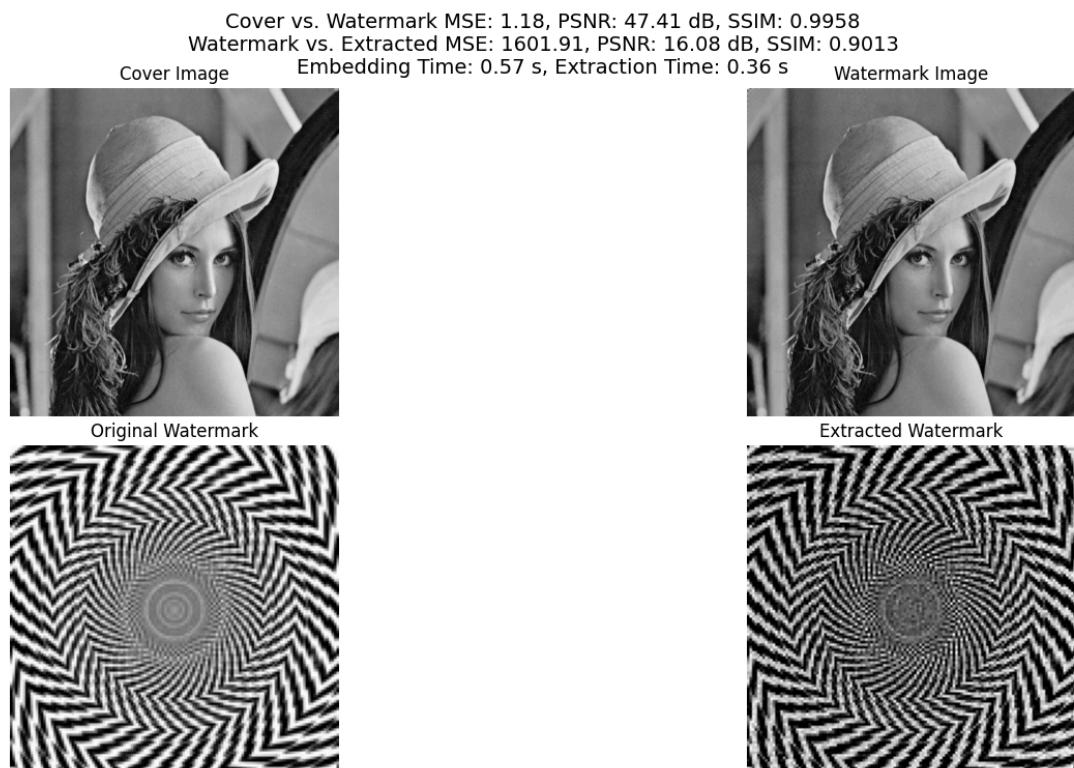
Original Secret Image (32x32)



Extracted Secret Image



## DCT - Low Frequency



Cover vs. Watermark MSE: 0.54, PSNR: 50.81 dB, SSIM: 0.9965  
Cover Image Watermark vs. Extracted MSE: 9583.53, PSNR: 8.32 dB, SSIM: 0.4768  
Embedding Time: 0.52 s, Extraction Time: 0.28 s Watermark Image



Università  
di Catania



Università  
di Catania

Original Watermark



Extracted Watermark



Cover vs. Watermark MSE: 1.45, PSNR: 46.53 dB, SSIM: 0.9959  
Watermark vs. Extracted MSE: 14901.99, PSNR: 6.40 dB, SSIM: 0.2669  
Cover Image Embedding Time: 0.43 s, Extraction Time: 0.28 s Watermark Image



Original Watermark



Extracted Watermark



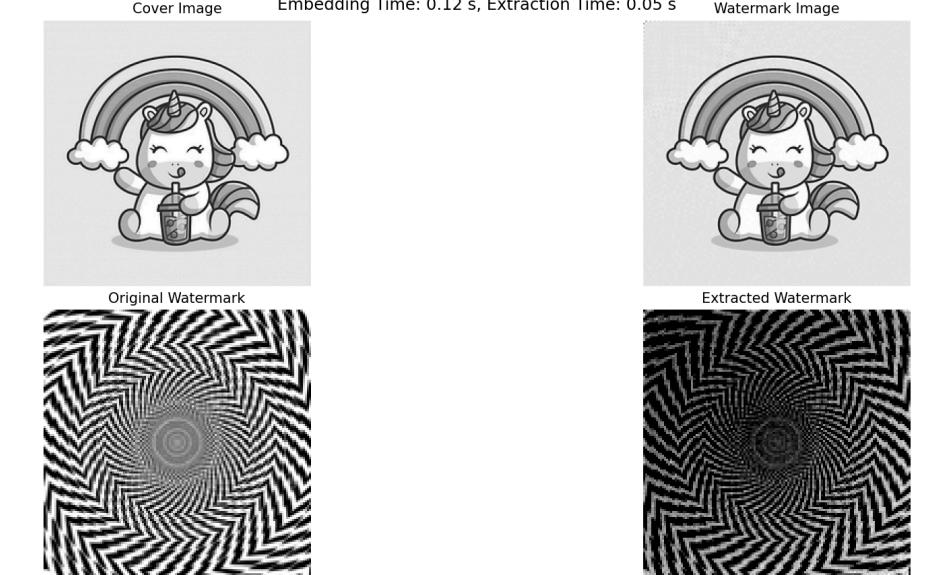
Cover vs. Watermark MSE: 3.02, PSNR: 43.33 dB  
Watermark vs. Extracted MSE: 25519.07, PSNR: 4.06 dB  
Embedding Time: 0.11 s, Extraction Time: 0.04 s



Watermark Image



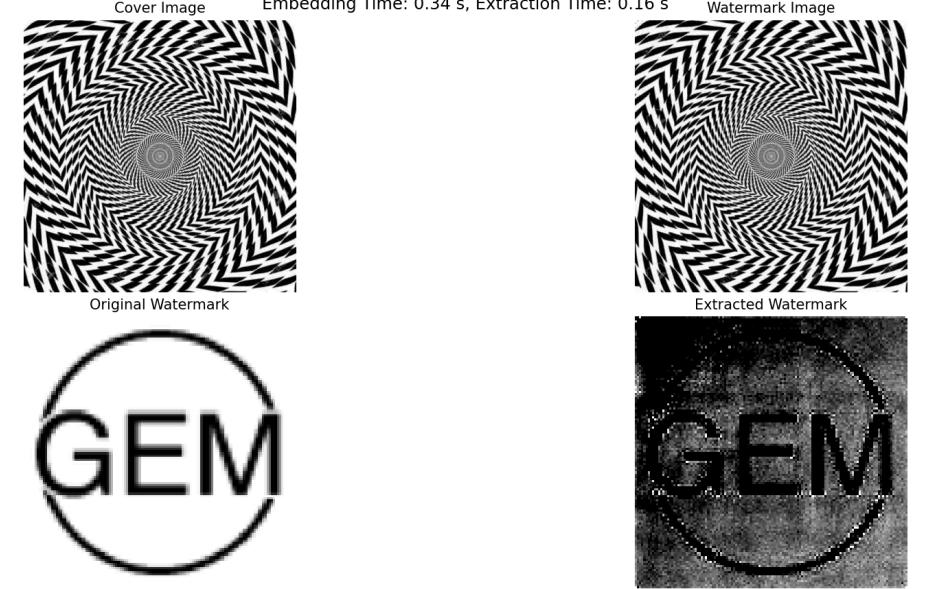
Cover vs. Watermark MSE: 2.45, PSNR: 44.24 dB  
Watermark vs. Extracted MSE: 6745.33, PSNR: 9.84 dB  
Embedding Time: 0.12 s, Extraction Time: 0.05 s



Watermark Image



Cover vs. Watermark MSE: 0.92, PSNR: 48.47 dB  
Watermark vs. Extracted MSE: 29332.20, PSNR: 3.46 dB  
Embedding Time: 0.34 s, Extraction Time: 0.16 s

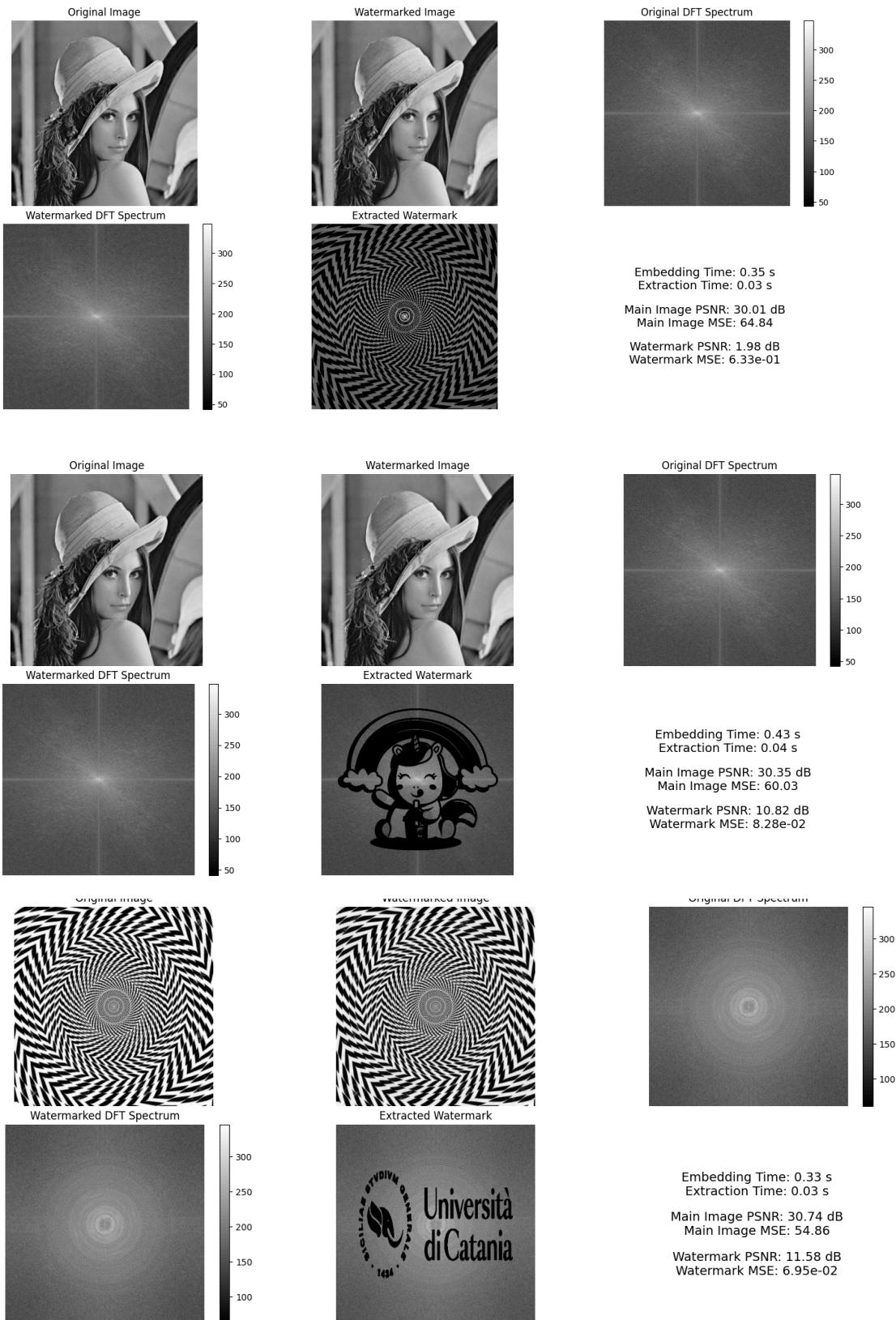


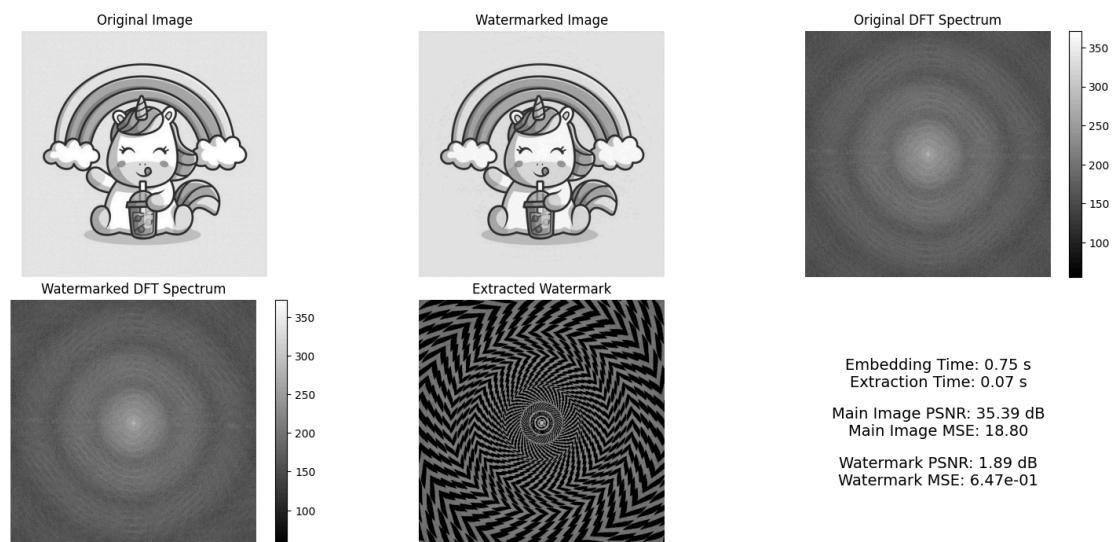
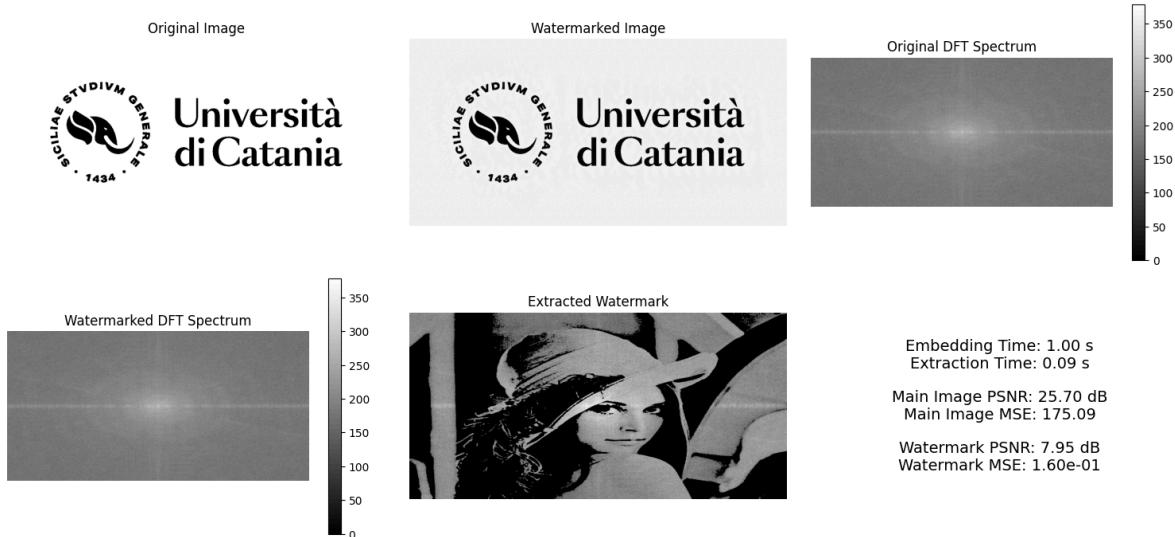
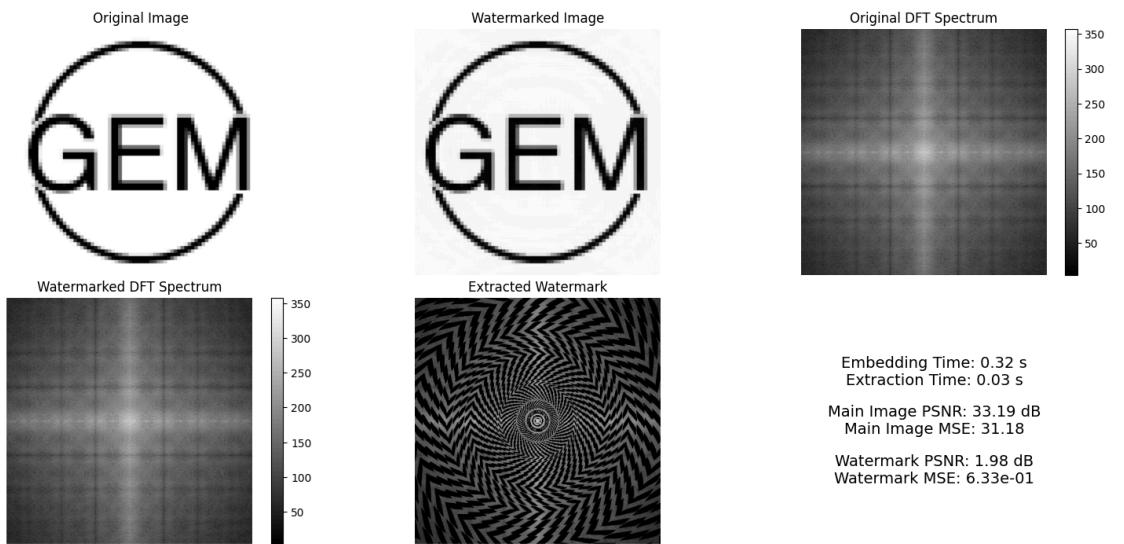
Original Watermark

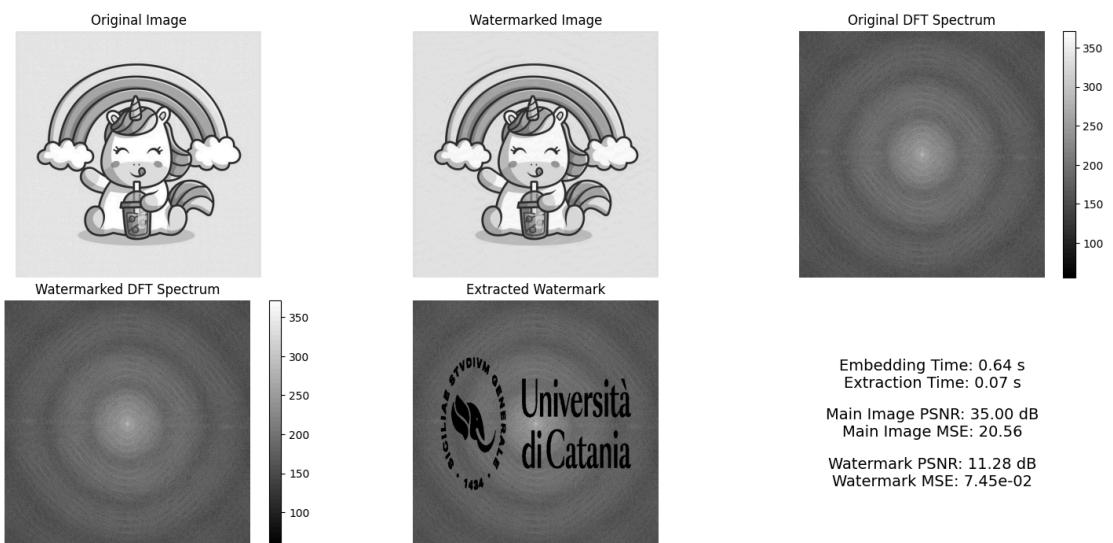
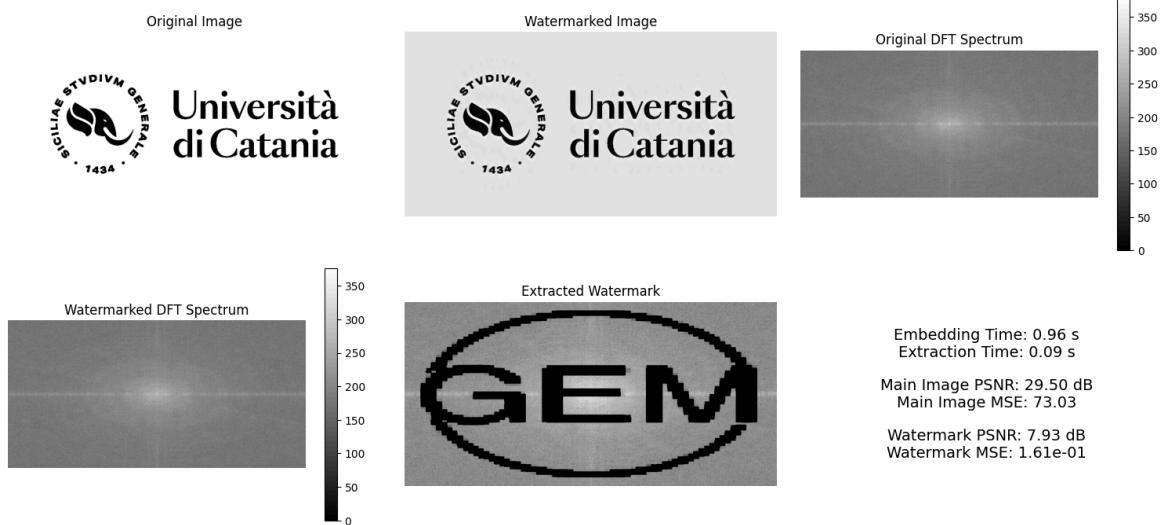


Extracted Watermark

## DFT

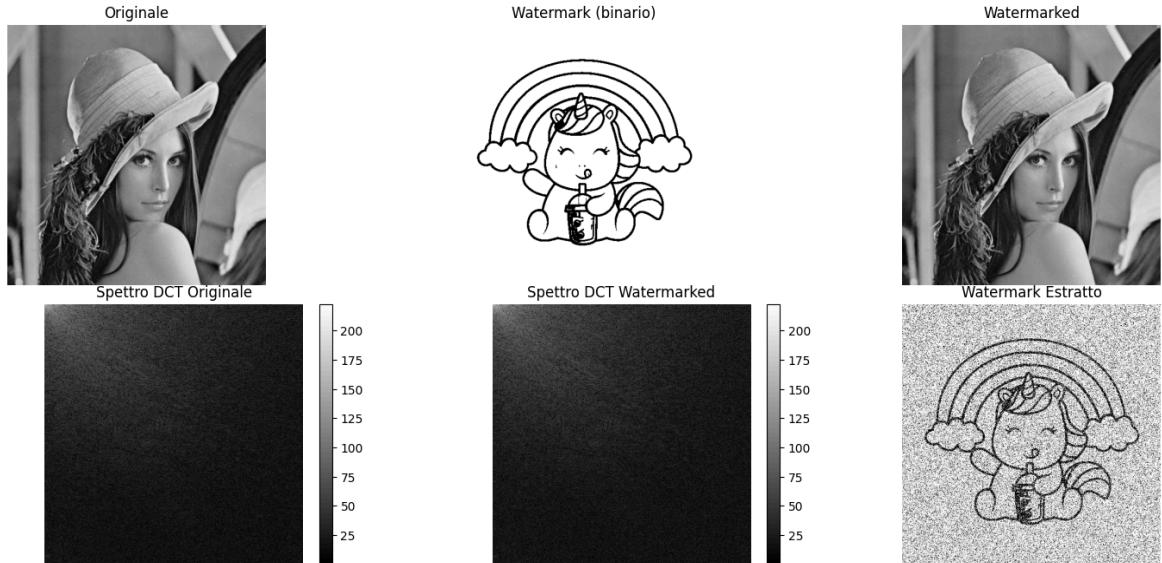




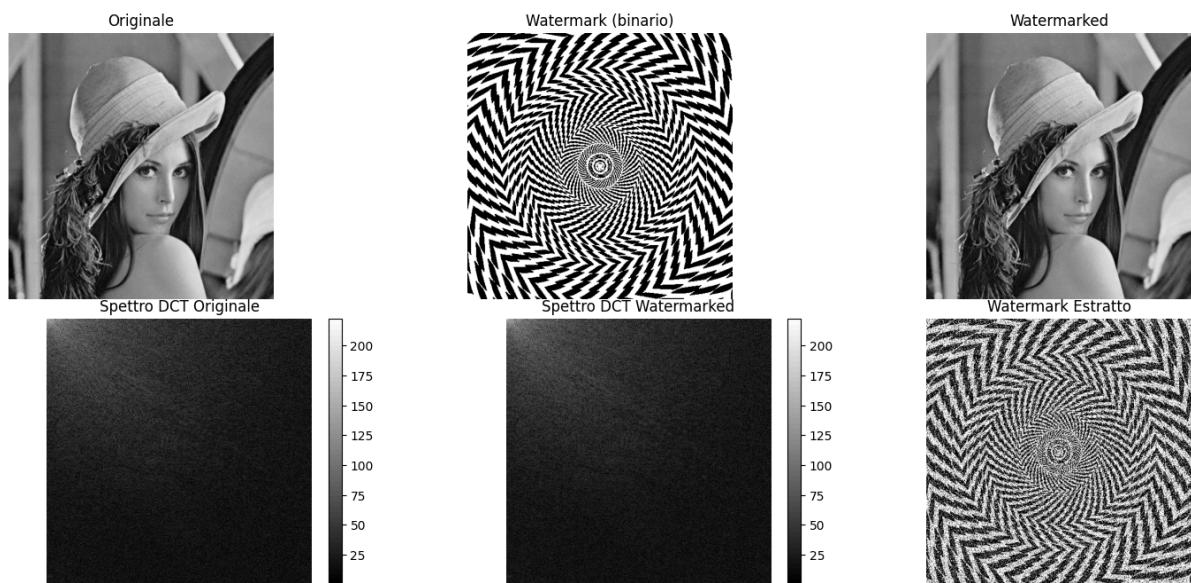


## DSSS

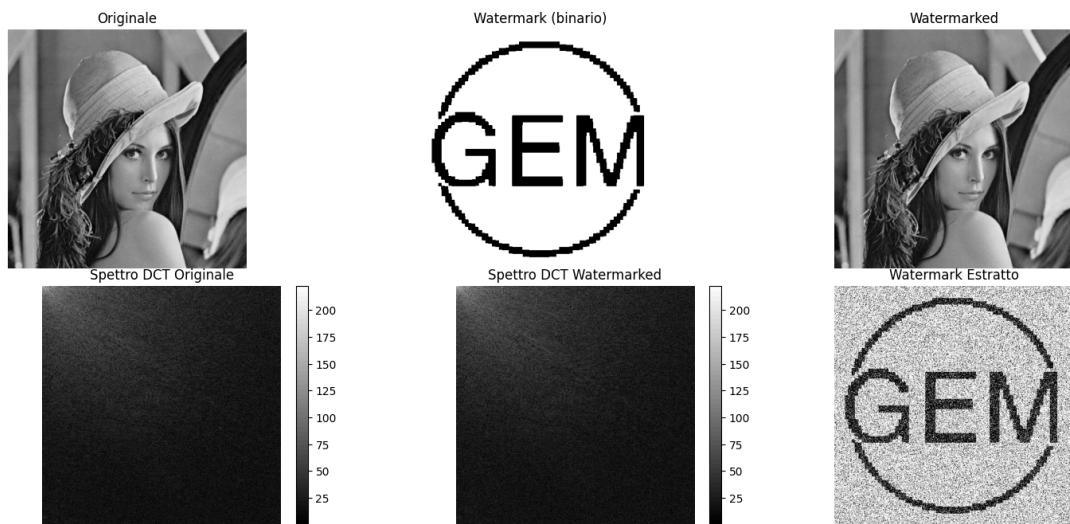
Main Image - PSNR: 47.21 dB, MSE: 1.24 | Watermark - PSNR: 8.34 dB, MSE: 1.46e-01  
 Embedding Time: 0.55 s, Extraction Time: 0.25 s



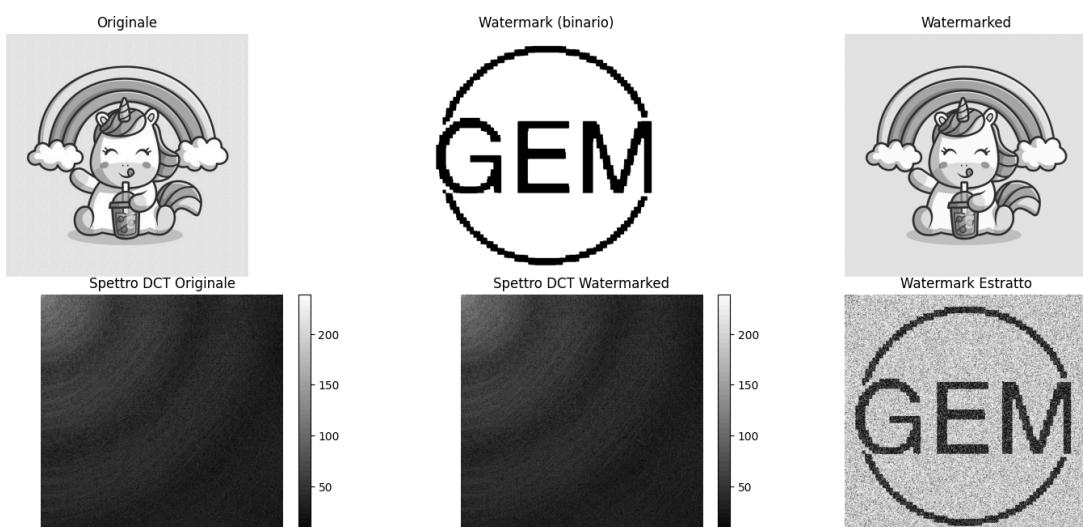
Main Image - PSNR: 48.94 dB, MSE: 0.83 | Watermark - PSNR: 8.31 dB, MSE: 1.48e-01  
 Embedding Time: 0.33 s, Extraction Time: 0.23 s



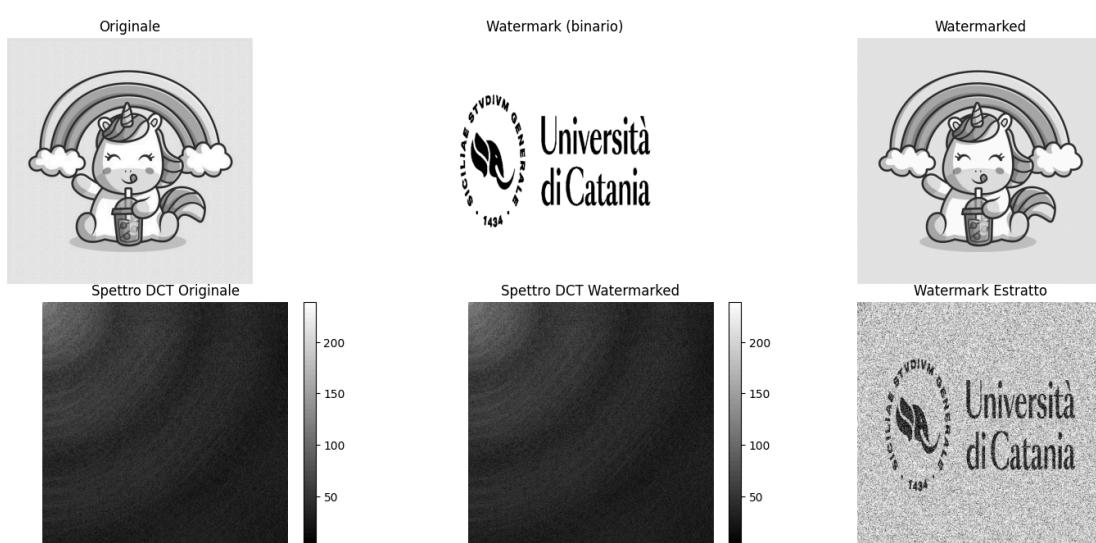
Main Image - PSNR: 47.46 dB, MSE: 1.17 | Watermark - PSNR: 8.31 dB, MSE: 1.47e-01  
 Embedding Time: 0.33 s, Extraction Time: 0.19 s



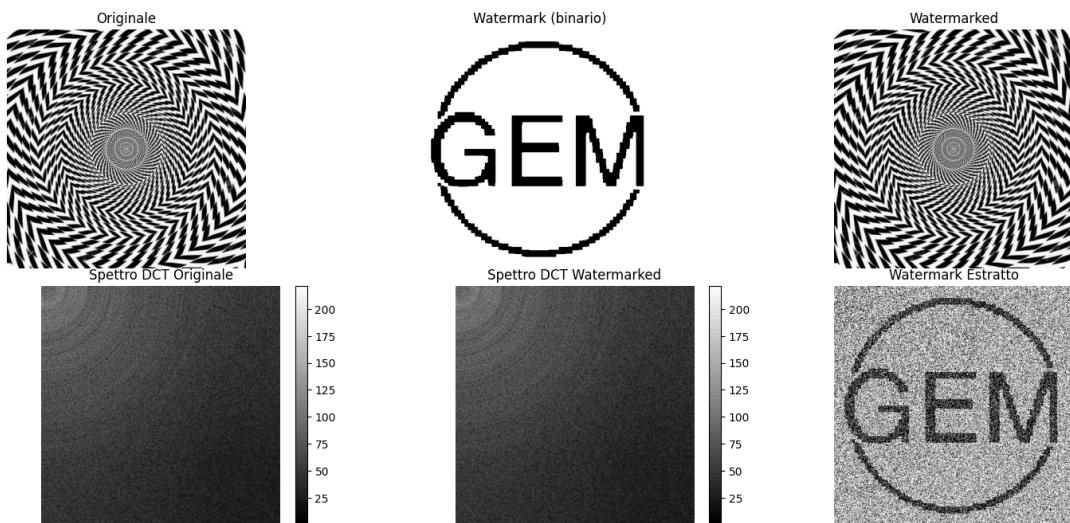
Main Image - PSNR: 47.48 dB, MSE: 1.16 | Watermark - PSNR: 8.15 dB, MSE: 1.53e-01  
 Embedding Time: 0.75 s, Extraction Time: 0.61 s



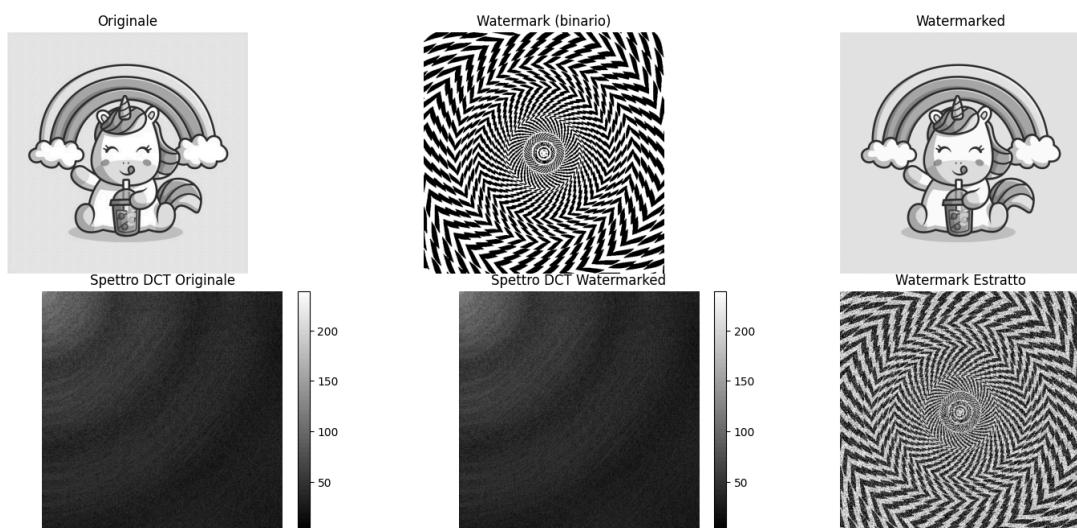
Main Image - PSNR: 47.15 dB, MSE: 1.25 | Watermark - PSNR: 8.11 dB, MSE: 1.55e-01  
 Embedding Time: 0.81 s, Extraction Time: 0.48 s



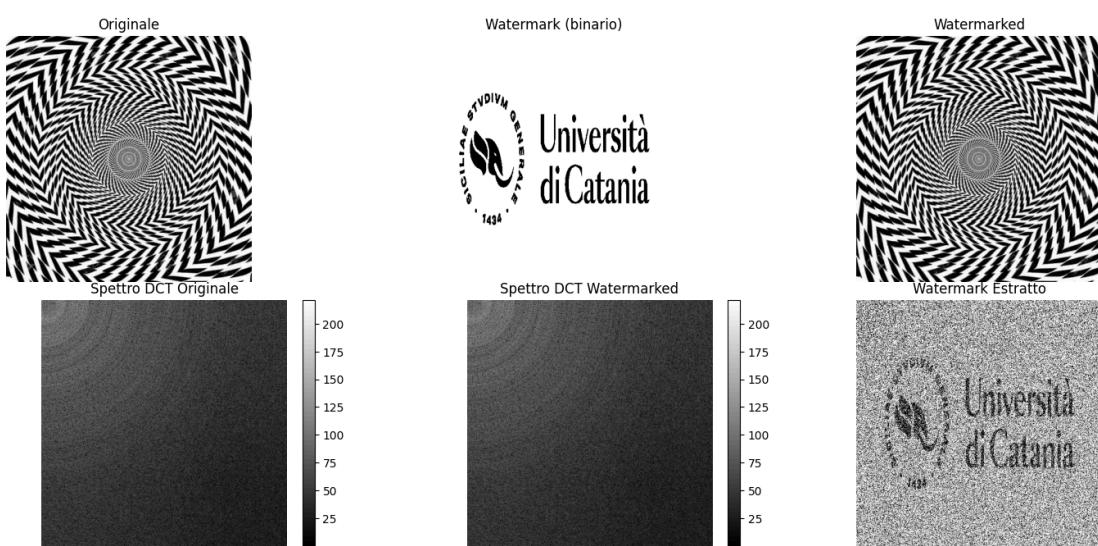
Main Image - PSNR: 48.25 dB, MSE: 0.97 | Watermark - PSNR: 6.11 dB, MSE: 2.45e-01  
 Embedding Time: 0.28 s, Extraction Time: 0.24 s



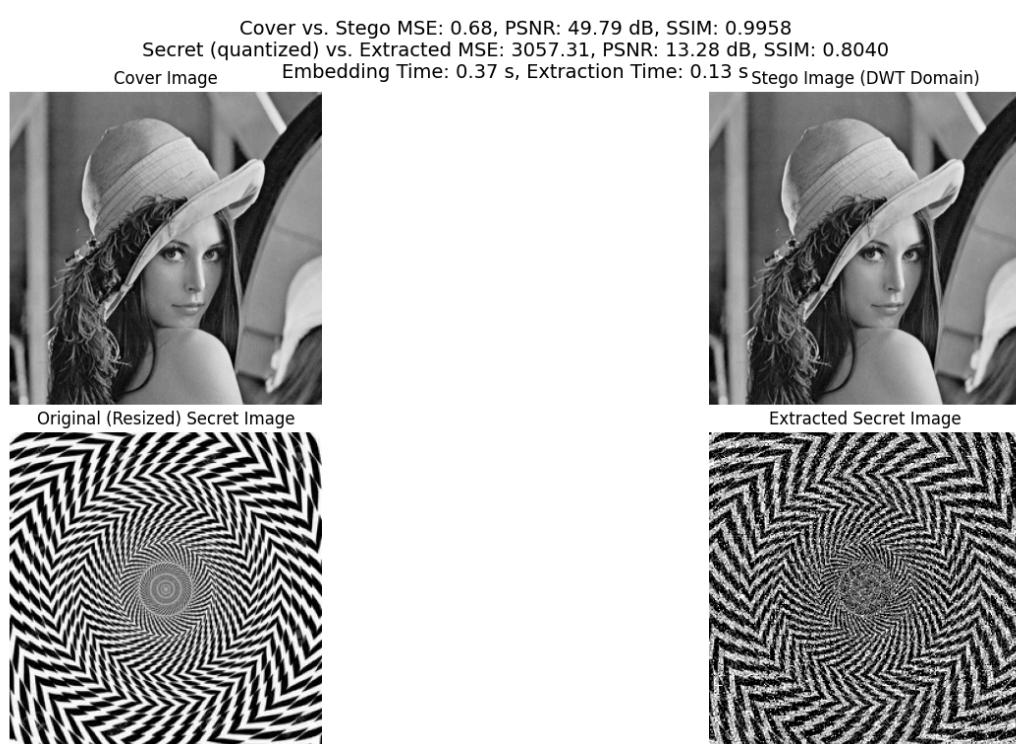
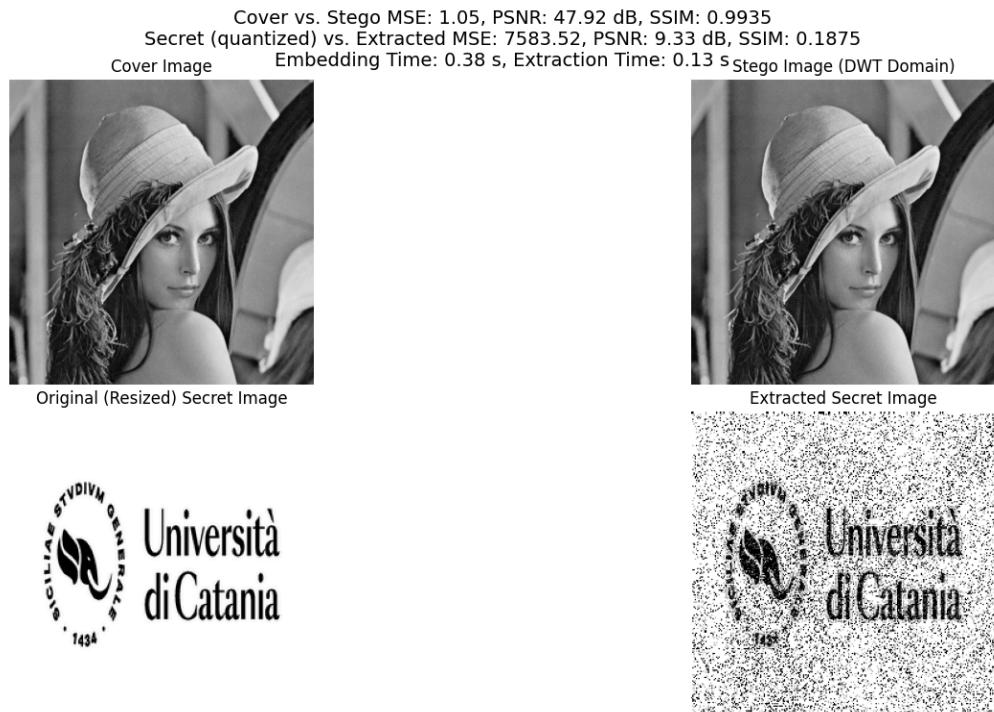
Main Image - PSNR: 48.96 dB, MSE: 0.83 | Watermark - PSNR: 8.26 dB, MSE: 1.49e-01  
 Embedding Time: 0.78 s, Extraction Time: 0.45 s



Main Image - PSNR: 47.92 dB, MSE: 1.05 | Watermark - PSNR: 5.95 dB, MSE: 2.54e-01  
 Embedding Time: 0.33 s, Extraction Time: 0.21 s



## DWT - One - Level



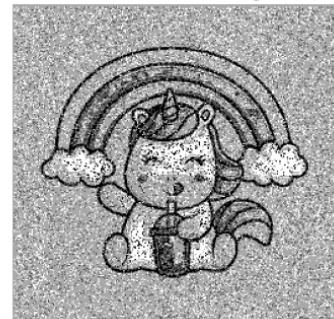
Cover vs. Stego MSE: 0.56, PSNR: 50.67 dB, SSIM: 0.9996  
Secret (quantized) vs. Extracted MSE: 2414.58, PSNR: 14.30 dB, SSIM: 0.3293  
Cover Image      Embedding Time: 0.43 s, Extraction Time: 0.11 s Stego Image (DWT Domain)



Original (Resized) Secret Image



Extracted Secret Image



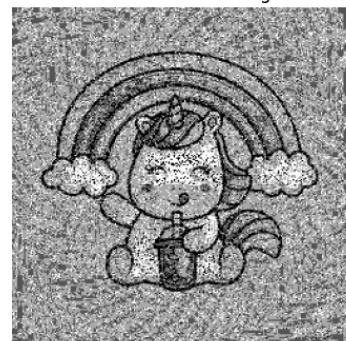
Cover vs. Stego MSE: 0.46, PSNR: 51.46 dB, SSIM: 0.9999  
Secret (quantized) vs. Extracted MSE: 3102.82, PSNR: 13.21 dB, SSIM: 0.3074  
Cover Image      Embedding Time: 0.34 s, Extraction Time: 0.10 s Stego Image (DWT Domain)



Original (Resized) Secret Image



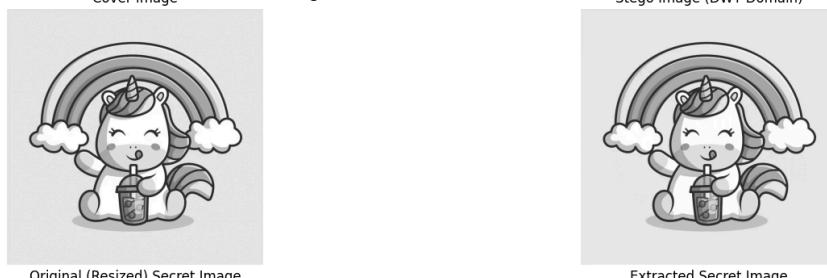
Extracted Secret Image



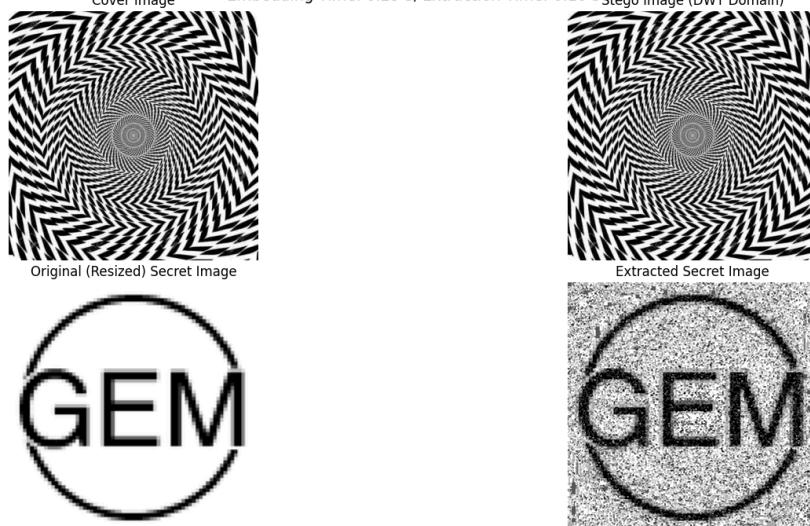
Cover vs. Stego MSE: 0.88, PSNR: 48.68 dB, SSIM: 0.9999  
 Secret (quantized) vs. Extracted MSE: 9054.02, PSNR: 8.56 dB, SSIM: 0.1828  
 Cover Image      Embedding Time: 0.45 s, Extraction Time: 0.13 s Stego Image (DWT Domain)



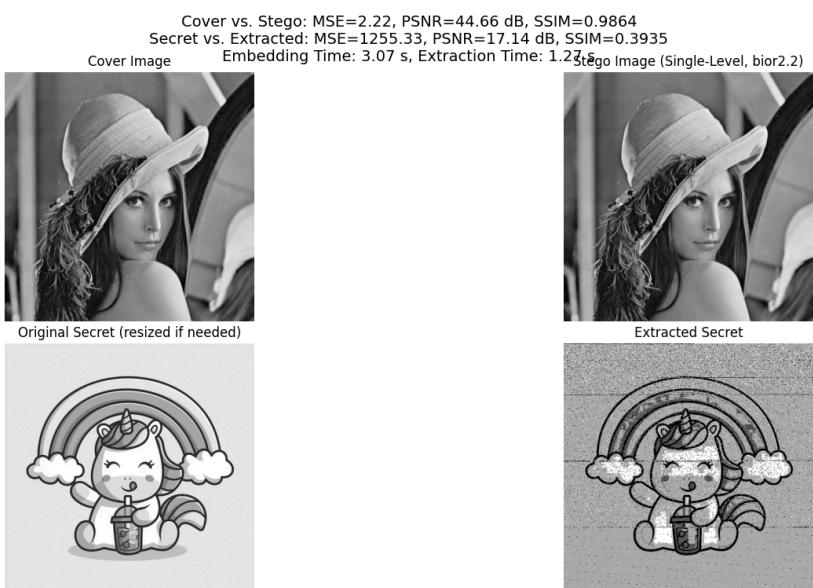
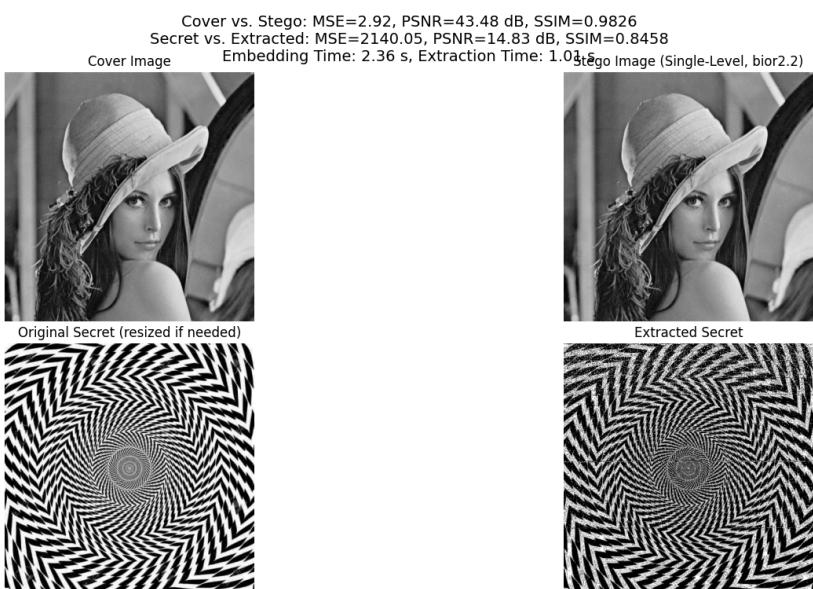
Cover vs. Stego MSE: 1.93, PSNR: 45.27 dB, SSIM: 0.9774  
 Secret (quantized) vs. Extracted MSE: 3985.10, PSNR: 12.13 dB, SSIM: 0.6886  
 Cover Image      Embedding Time: 0.56 s, Extraction Time: 0.16 s Stego Image (DWT Domain)



Cover vs. Stego MSE: 0.82, PSNR: 49.00 dB, SSIM: 0.9999  
 Secret (quantized) vs. Extracted MSE: 7857.79, PSNR: 9.18 dB, SSIM: 0.2845  
 Cover Image      Embedding Time: 0.29 s, Extraction Time: 0.10 s Stego Image (DWT Domain)



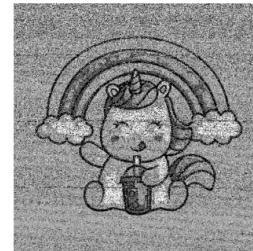
## DWT - MultiLevel



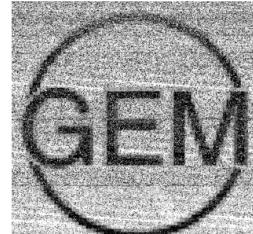
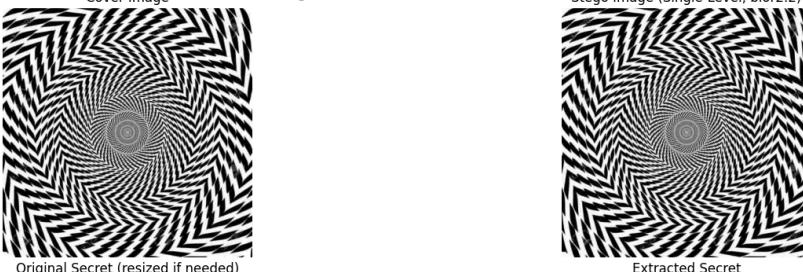
Cover vs. Stego: MSE=2.35, PSNR=44.42 dB, SSIM=0.9711  
Secret vs. Extracted: MSE=5701.03, PSNR=10.57 dB, SSIM=0.2212  
Cover Image      Embedding Time: 3.12 s, Extraction Time: 2.37 s      Stego Image (Single-Level, bior2.2)



Cover vs. Stego: MSE=1.87, PSNR=45.42 dB, SSIM=0.9998  
Secret vs. Extracted: MSE=4157.14, PSNR=11.94 dB, SSIM=0.1989  
Cover Image      Embedding Time: 2.50 s, Extraction Time: 1.29 s      Stego Image (Single-Level, bior2.2)



Cover vs. Stego: MSE=3.46, PSNR=42.74 dB, SSIM=0.9996  
Secret vs. Extracted: MSE=11085.22, PSNR=7.68 dB, SSIM=0.1216  
Cover Image      Embedding Time: 2.14 s, Extraction Time: 0.93 s      Stego Image (Single-Level, bior2.2)



Cover vs. Stego: MSE=1.06, PSNR=47.88 dB, SSIM=0.9896  
Cover Image Secret vs. Extracted: MSE=12890.32, PSNR=7.03 dB, SSIM=0.0883  
Embedding Time: 5.15 s, Extraction Time: 2.86 s<sup>Stego Image (Single-Level, bior2.2)</sup>



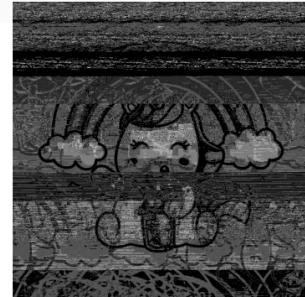
Università  
di Catania

Original Secret (resized if needed)



Università  
di Catania

Extracted Secret

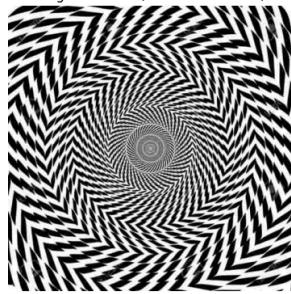


Cover vs. Stego: MSE=0.47, PSNR=51.43 dB, SSIM=0.9947  
Cover Image Secret vs. Extracted: MSE=8093.27, PSNR=9.05 dB, SSIM=0.4441  
Embedding Time: 2.68 s, Extraction Time: 2.86 s<sup>Stego Image (Single-Level, bior2.2)</sup>



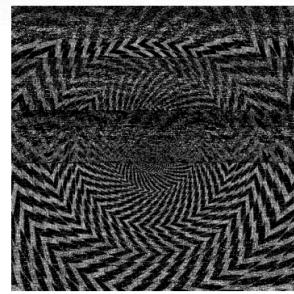
Università  
di Catania

Original Secret (resized if needed)



Università  
di Catania

Extracted Secret

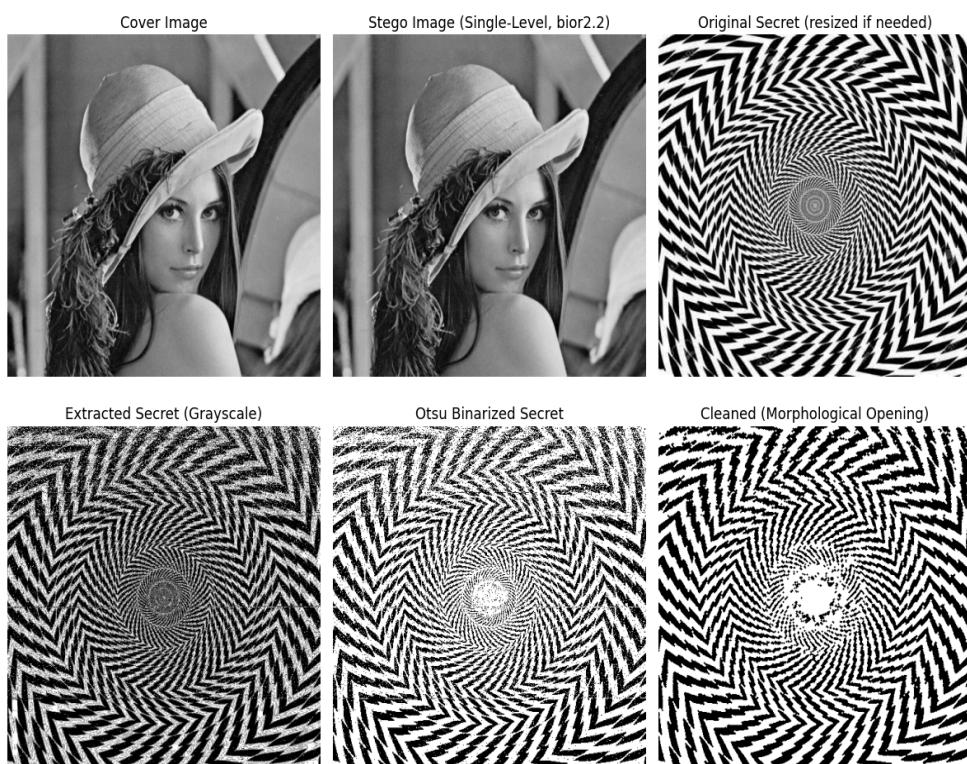


## DWT - Otsu

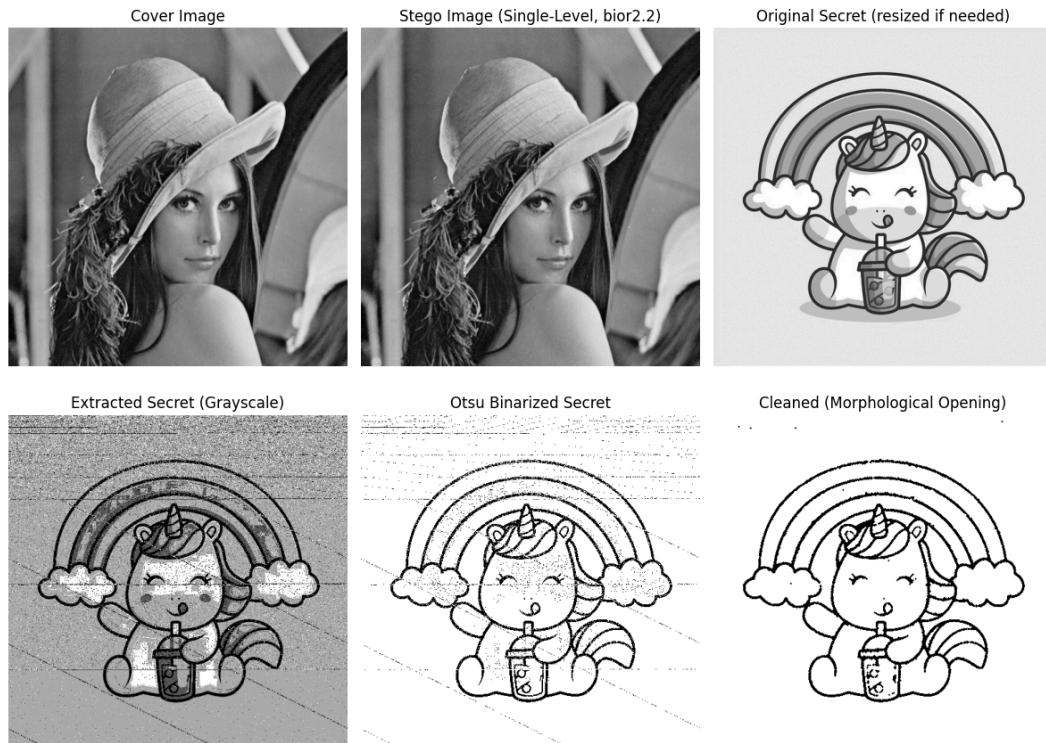
Cover vs. Stego: MSE=4.82, PSNR=41.30 dB, SSIM=0.9723  
 Secret vs. Extracted: MSE=1640.55, PSNR=15.98 dB, SSIM=0.7808  
 Embedding Time: 1.13 s, Extraction Time: 0.50 s



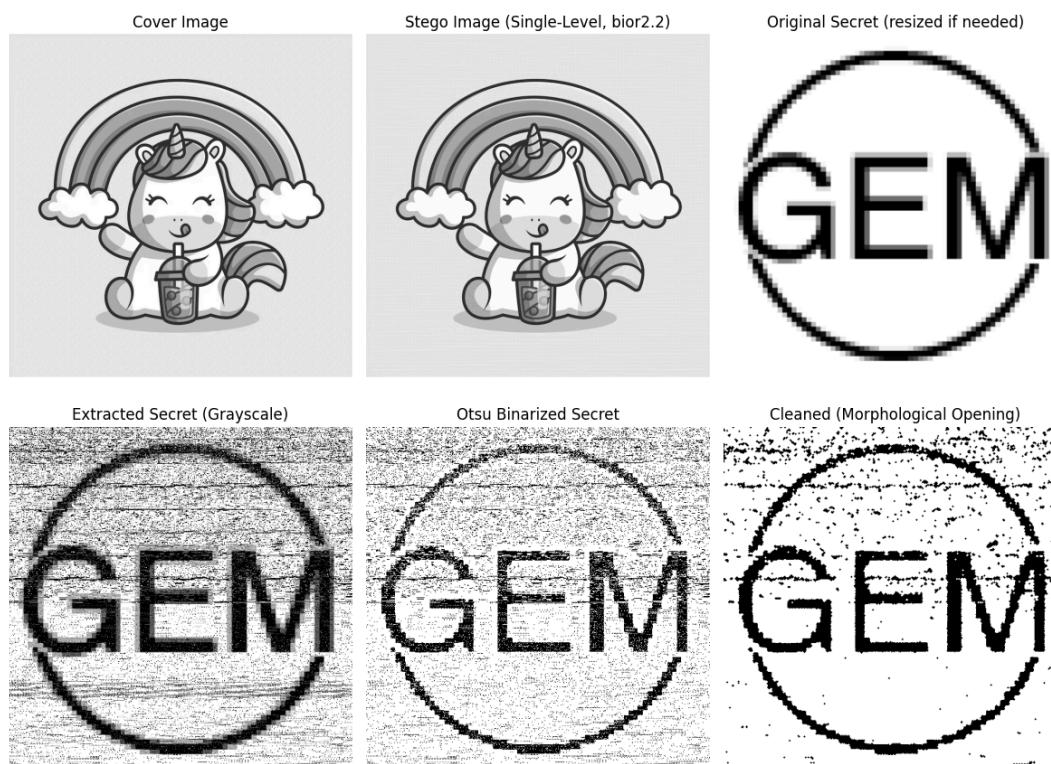
Cover vs. Stego: MSE=2.76, PSNR=43.72 dB, SSIM=0.9833  
 Secret vs. Extracted: MSE=7355.19, PSNR=9.46 dB, SSIM=0.6738  
 Embedding Time: 1.04 s, Extraction Time: 0.50 s



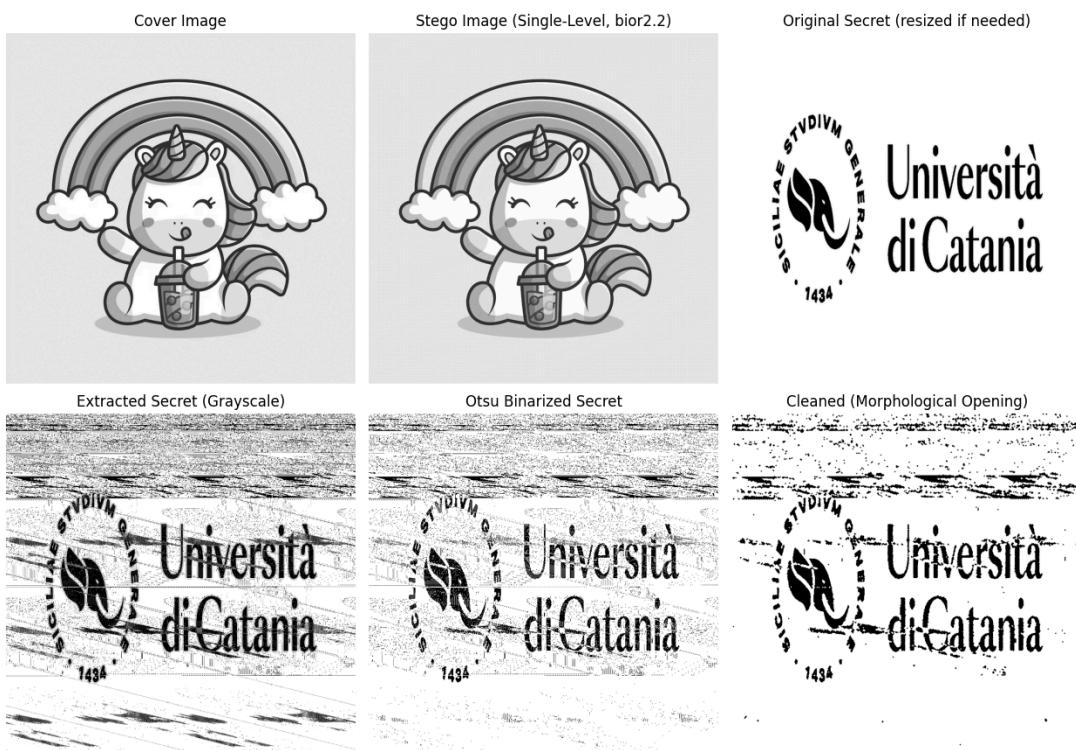
Cover vs. Stego: MSE=2.26, PSNR=44.58 dB, SSIM=0.9862  
 Secret vs. Extracted: MSE=7745.88, PSNR=9.24 dB, SSIM=0.7783  
 Embedding Time: 1.16 s, Extraction Time: 0.50 s



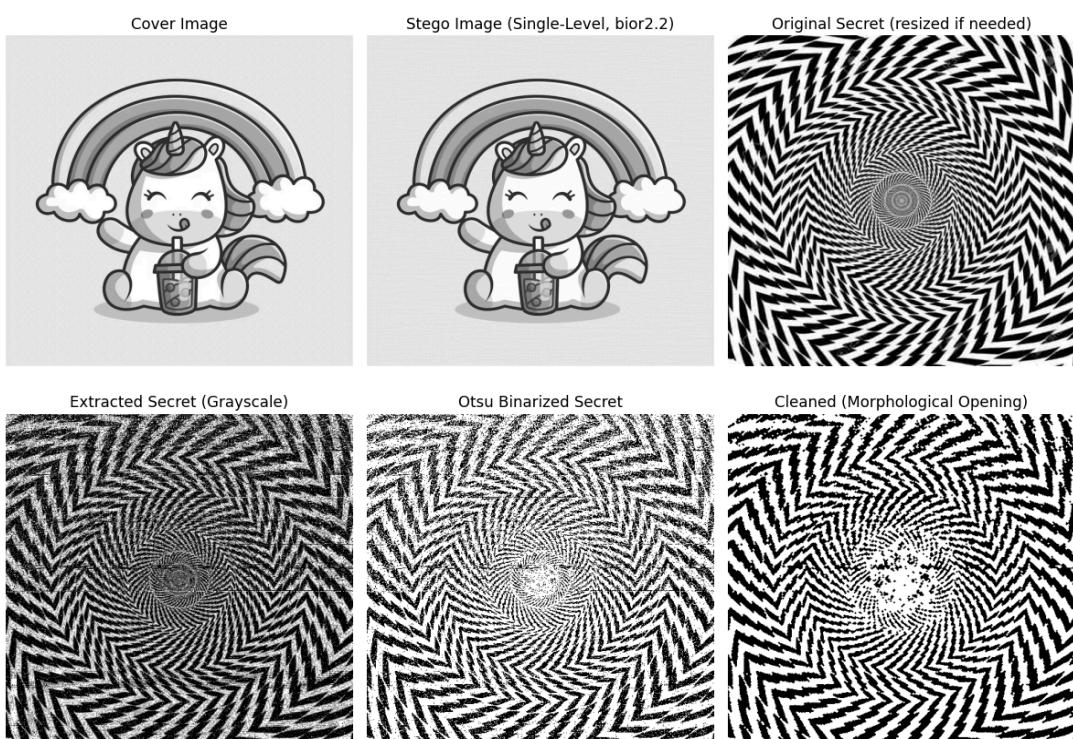
Cover vs. Stego: MSE=2.17, PSNR=44.77 dB, SSIM=0.9746  
 Secret vs. Extracted: MSE=4685.40, PSNR=11.42 dB, SSIM=0.5054  
 Embedding Time: 1.23 s, Extraction Time: 1.00 s



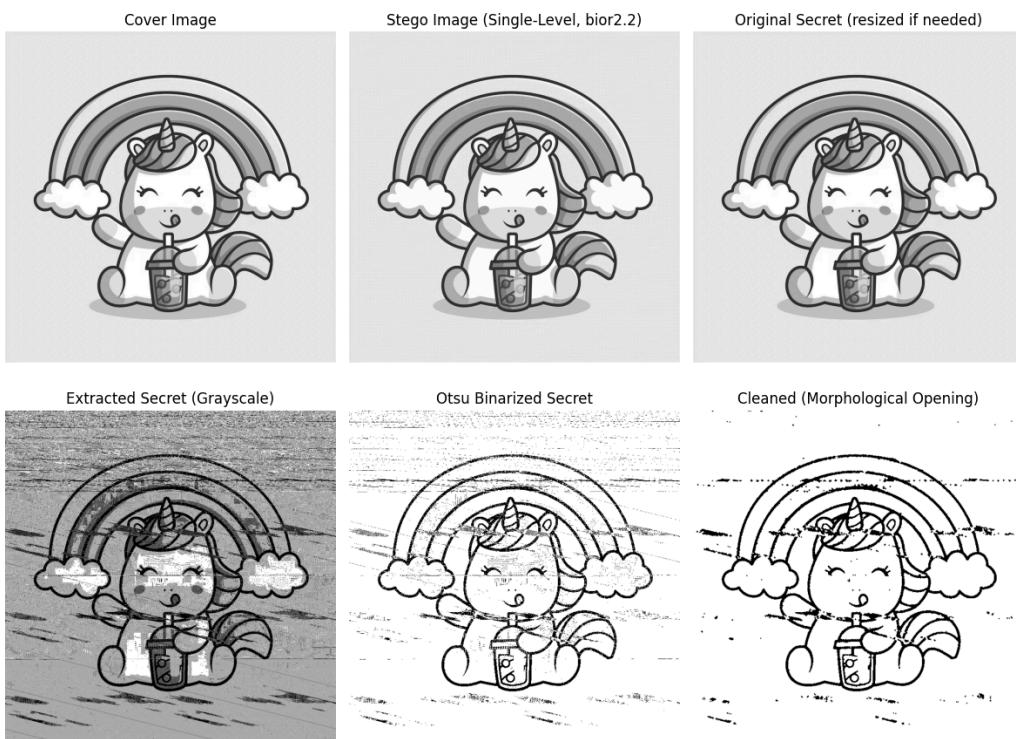
Cover vs. Stego: MSE=6.13, PSNR=40.26 dB, SSIM=0.9323  
 Secret vs. Extracted: MSE=3488.60, PSNR=12.70 dB, SSIM=0.7304  
 Embedding Time: 2.14 s, Extraction Time: 1.02 s



Cover vs. Stego: MSE=1.06, PSNR=47.88 dB, SSIM=0.9888  
 Secret vs. Extracted: MSE=7832.39, PSNR=9.19 dB, SSIM=0.6500  
 Embedding Time: 1.22 s, Extraction Time: 1.00 s



Cover vs. Stego: MSE=2.95, PSNR=43.43 dB, SSIM=0.9652  
Secret vs. Extracted: MSE=7998.34, PSNR=9.10 dB, SSIM=0.7346  
Embedding Time: 2.15 s, Extraction Time: 1.01 s



# Conclusione

La diversità degli approcci adottati ha evidenziato come non esista una soluzione universale: ogni tecnica presenta vantaggi specifici che possono essere maggiormente o meno adatti a differenti scenari applicativi.

Mentre metodi come la DWT e la DCT offrono un buon bilanciamento tra robustezza e qualità visiva, tecniche come LSB risultano più accessibili ma con limitazioni in termini di sicurezza. Allo stesso modo, l'uso di trasformate come la DFT nel watermarking mostra come il dominio della frequenza possa essere sfruttato per garantire una protezione più durevole, mentre metodi come lo spread spectrum offrono resistenza a manipolazioni esterne.

Questa analisi ci ha permesso di comprendere meglio le potenzialità e le criticità di ciascun metodo, fornendo spunti utili per applicazioni future in cui la scelta della tecnica dipenderà dalle specifiche esigenze di protezione e dalle caratteristiche del contenuto da tutelare.