Image Watermarking

Marcu Ioan

Năstase Marius

Grupa: 461

CUPRINS

1. Problema abordată în proiect.………………………..3

2. Justificarea problemei abordate……………………...4

3. Abordarea tehnică……………………………………4-6

4. Tehnologii……………………………………………6-10

5.1 Rezultate-SVD……………………………………...11-12

5.2 Rezultate-DWT……………………………………..13-14

6. Concluzii……………………………………………..14-15

7. Bibliografie…………………………………………..16

1. **Problema abordata in proiect**

Imaginile sunt rapid și ușor distribuite și reproduse pe internet, această accesibilitate extinsă, deși benefică pentru partajarea informațiilor, prezintă un risc semnificativ pentru drepturile de autor și integritatea imaginilor digitale. Pentru această problemă ne sare în ajutor Watermarking-ul, care este o tehnică de încorporare a unei semnături invizibile sau vizibile într-o imagine.

Watermarking-ul ajută la:

* *Protecția drepturilor de autor* (prevenirea utilizării neautorizate, drepturi economice – exemplu: proprietarii pot beneficia din punct de vedere economic de pe urma unei imagini)
* *Menținerea integrității imaginii* (prevenirea alterării, trasabilitate – watermark-ul ajută la urmărirea surselor și stabilirea responsabilităților pentru a pune în evidență încălcarea drepturilor de autor)

Integrarea **SVD (Singular Value Decomposition)** și **DWT (Discrete Wavelet Transform)** cu **Teoria Informației:** Proiectul combină aceste tehnici puternice pentru a dezvolta un sistem de watermarking eficient și robust. Prin aplicarea principiilor Teoriei Informației, se alege locația optimă pentru watermark, în timp ce SVD și DWT sunt folosite pentru a asigura că watermark-ul este atât invizibil, cât și rezistent la diferite forme de manipulare digitală.

**2. Justificarea problemei abordate**

Aplicarea watermark-ului se realizează cu ajutorul a doi algoritmi și anume prin Singular Value Decomposition, care permite modificarea componentelor esențiale ale imaginii pentru a încorpora watermark-ul, asigurând un echilibru între invizibilitate și robustețe, iar al doilea algoritm este Discrete Wavelet Transform facilitează localizarea zonei optime în imagine pentru a insera watermark-ul, bazându-se pe caracteristicile informaționale ale imaginii.

Poziționarea watermark-ului se calculează cu ajutorul Teoriei Informației, în care calculăm zona cu entropia cea mai mare pentru a poziționa watermark-ul, fapt prin care maximizăm invizibilitatea și minimizăm impactul asupra calității imaginii.

Această tehnică poate fi aplicată in diverse domenii precum:

* *Artă Digitală -* unde opera este adesea vulnerabilă la furt și utilizare neautorizată
* *Documente Corporative -* unde confidențialitatea și protecția identității de brand sunt esențiale
* *Media Digitală –* aplicarea watermark-ului pe videoclipuri și transmisiuni online

**3. Abordarea tehnică**

***Entropia Shannon:***

Pentru identificarea zonei optime de plasare a watermark-ului, se utilizează Entropia Shannon. Entropia unui bloc este calculată astfel:

reprezintă probabilitatea de apariție a valorilor pixelilor în blocul B, astfel măsoară impredictibilitatea informațiilor din bloc pentru a aplica watermark-ul

***Descompunerea în Valori Singulare (SVD):***

***a) Aplicare SVD***

O imagine este reprezentată ca o matrice A, iar SVD este aplicată pentru a descompune această matrice în trei componente.

și sunt două matrici ortogonale

este o matrice diagonal care conține volorile singulare

***b) Încorporarea Watermark-ului***

Valorile singulare Σ sunt apoi ajustate cu informația watermark-ului.

reprezintă watermark-ul

(alpha) este factorul de scalare

***Transformata Discretă Wavelet (DWT):***

***a) Aplicare DWT***

DWT este folosită pentru a descompune imaginile în diferite niveluri de frecvență și detalii, iar pentru un nivel de descompunere:

reprezintă componenta cu frecvențe joase

reprezintă detalii la diferite orientări și frecvențe

***b) Plasarea Watermark-ului în LL***

Entropia blocurilor din sub-banda LL este calculată, iar watermark-ul este plasat în blocul cu cea mai mare entropie pentru a minimiza impactul asupra calității imaginii.

***Reconstrucția Imaginii:***

**SVD:** După ajustarea valorilor singulare cu watermark-ul, imaginea este reconstruită utilizând componentele modificate:

**DWT:** Imaginea este, de asemenea. reconstruită din coeficienții DWT modificați, reasamblând sub-benzile pentru a obține imaginea finală watermark-uită.

**4. Tehnologii**

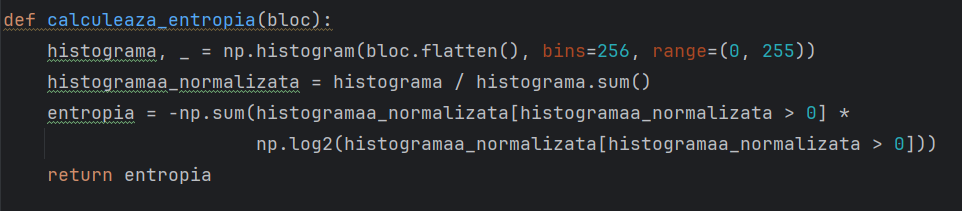
Bibliotecile folosite:

* **NumPy** pentru procesarea datelor numerice și operații matematice
* **PyWavelets (pywt)** utilizat pentru a aplica DWT asupra imaginilor, de asemenea, pentru descompunerea imaginilor în sub-benzi de frecvență și integrarea watermark-ului
* **Pillow (PIL)** utilizat pentru încărcarea și salvarea imaginilor
* **Scikit-Image (skimage)** utilizat pentru operațiuni precum divizarea imaginii în blocuri și calculul entropiei

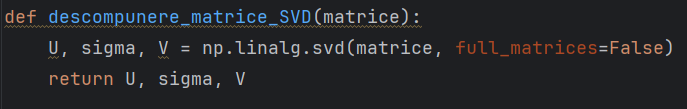
**Algoritmul SVD:**

Începem prin calcularea Entropiei Shannon pentru un bloc.

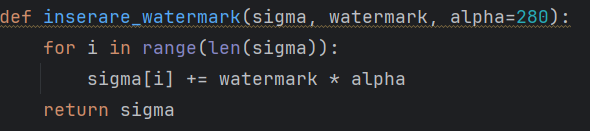
Calculăm histograma valorilor pixelilor folosind numpy, iar cu ajutorul .flattern() - transformăm blocul (care poate fi 2D sau 3D) într-un vector



După calcularea entropiei pentru un bloc, trebuie să găsim blocul cu cea mai mare entropie unde trebuie plasat watermark-ul, iar pentru asta ne folosim de funcția “gaseste\_bloc\_cu\_entropie\_maxima” în care întâi se verifică dacă am ales corect marimea pentru un bloc, iar apoi se parcurge imaginea pe blocuri și se compară entropia pentru bloc cu entropia maxima și actualizăm entropia maximă, respectiv coordonatele acesteia, dacă este cazul. De menționat că imaginea este transformată într-o matrice.

Descompunem valorile singulare ale matricii corespunzatoare imaginii încărcate, am setat parametrul "full\_matrices=False" pentru ca dimensiunile matricilor U și V să fie reduse pentru a se potrivi cu numărul de valori singulare - len(sigma).

În continuare o să introducem watermark-ul în imagine prin modificarea valorilor din vectorul sigma, practic adăugăm înmulțirea "watermark \* alpha" la fiecare valoare din sigma.

Urmează reconstruirea imaginii cu sigma-ul modificat, în care transformăm vectorul sigma modificat într-o matrice diagonală, de asemenea, ajutsăm dimensiunea imaginii la bloc, după care vom introduce wartermark-ul în imagine, mai exact în funcția “introducere\_watermark\_folosind\_SVD\_entropie” în care se aplică SVD pe fiecare canal de culoare al blocului, iar procesarea blocurilor se face corespunzător culorilor RGB.

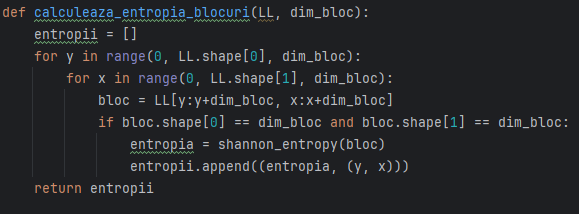
**Algoritmul DWT:**

Pentru o abordare secundară a temei propuse, am utilizat metoda DWT. Principiul de funționare a acestei metode este de a descompune imaginea originală în mai multe sub-benzi de frecvență, benzi ce se categorisesc astfel:

* Apropiere LL (Low-Low) - componenta de frecvență joasă a imaginii
* Orizontală LH (Low-High)
* Verticală HL (High-Low)
* Diagonală HH (High-High)

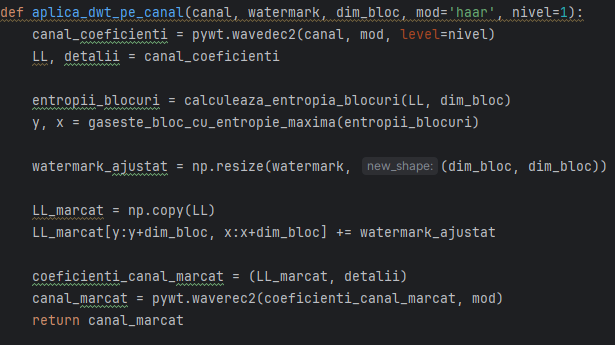
Pentru a introduce watermark-ul, vom modifica coeficienții componentei de frecvență joasă a imaginii, iar la sfârșit vom aplica transformarea inversă wavelet pentru a reconstrui imaginea modificată.

Începem prin calcularea entropiei pentru un bloc de mărime dată. Această căutare o facem în partea imaginii cu frecvențe joase (LL). Primul pas este acela de a calcula entropiile (pentru calculul entropiilor, în acest caz, am folosit funția ”shannon\_entropy” din scipy.stats.entropy) pentru toate blocurile din LL:



Următorul pas a fost să extragem blocul cu entropie maximă, pentru acest lucru am sortat descrescător lista cu entropii, rezultată în urma aplicării funcției de mai sus și am extras primul element.

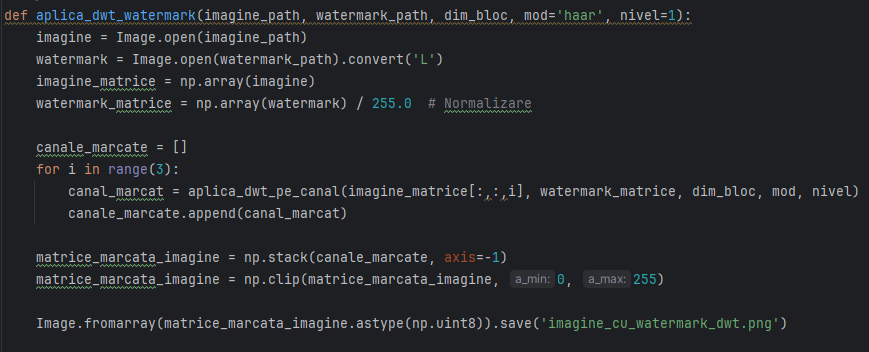
Am continuat prin crearea unei funcții care să introducă watermark-ul nostru în blocul cu entropie maximă. În această funcție am implementat logica de descompunere a imaginii în sub-benzi de frecvență, am calculat entropiile pentru blocurile ce formează imaginea și am extras blocul cu entropie maximă, unde vom insera watermark-ul. Acest bloc modificat va fi returnat la final:



În ultimă fază, am creat funcția următoare ce adaugă watermark-ul în imaginea selectată de noi. Funcția are ca parametrii:

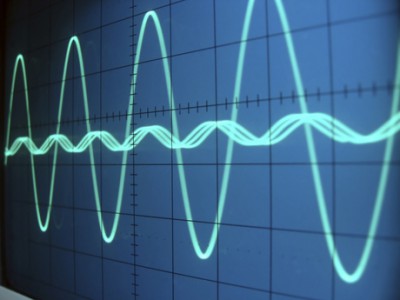
* Path-ul către imaginea unde vom insera watermark-ul: imagine\_path
* Path-ul către imaginea ce reprezintă watermark-ul: watermark\_path
* Dimensiunea blocului din imagine unde va fi inserat watermark-ul: dim\_bloc
* Utilizarea wavelet-ului haar ca funcție pentru descompunerea și analiza imaginii: mod=`haar`
* Setarea nivelului de descompunere la 1: nivel=1

Logica funcției:

* Am încărcat imaginea și watermark-ul de la path-urile specificate;
* Am transformat imaginile încărcate în matrici;
* Am aplicat watermark-ul pentru fiecare canal de culoare;
* Am recombinat canalele modificate și le-am normalizat;
  + np.stack(canale\_marcate, axis=-1) - recombină canalele marcate înapoi într-o singură imagine;
  + np.clip(matrice\_marcata\_imagine,0,255) - normalizează, se asigură că toate valorile pixelilor din matrice sunt între 0 și 255.
* La final, salvăm imaginea nouă.

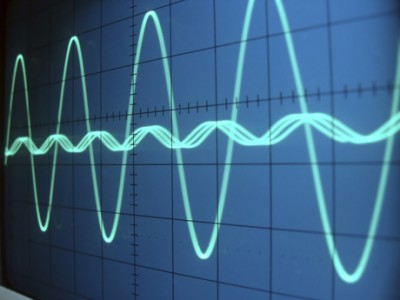
**5.1 Rezultate – SVD**

Pentru testarea implementării prezentate am selectat următoarea imagine în care vom insera watermark-ul. Am configurat în cod dimensiunea blocului în care vom introduce watermark-ul.



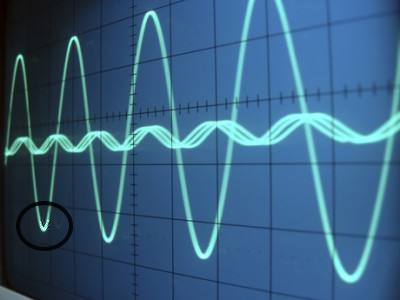
Pentru implemtarea algoritmului cu SVD trebuie să alegem cu grijă valorile pentru alpha, respectiv watermark. În acest sens, am rulat programul pentru valori diferite ale parametrului alpha (de menționat ca valoarea watermark-ului este constant 1):

1. alpha = 0.005, mărime bloc = 10

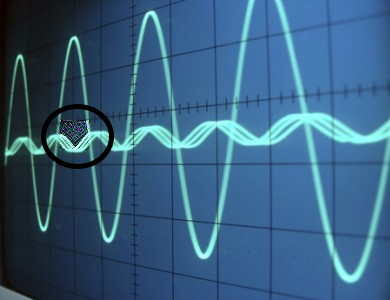


Putem observa că pentru aceste valori, calitatea imaginii nu a fost afectată.

b) alpha = 140, mărime bloc = 10



După o creștere mare a parametrului alpha, se poate observa că imaginea a fost alterată în blocul cu watermark.

c) alpha = 280 (pentru vizibilitate), mărime bloc = 30

Pentru a evidenția faptul că poziționăm watermark-ul în blocul cu entropia maximă, am ajusatat parametrii alpha și mărime bloc astfel încât să se poată observa că locația watermark-ului s-a modificat.

**5.2 Rezultate – DWT**

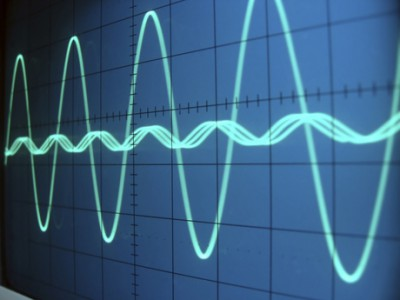
Pentru testarea implementării prezentate am selectat aceeași imagine de mai sus în care vom insera watermark-ul (reprezentat de o altă imagine).



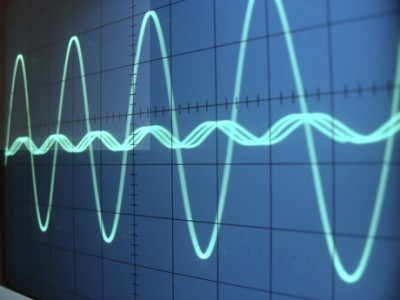
(watermark)

Pentru implemtarea algoritmului cu DWT trebuie să avem grijă la normalizarea watermark-ului, în cazul nostru valorile matricei care reprezintă watermark-ul au fost împărțite la 255 (valoarea maximă a unui pixel), pentru evitarea alterării imaginii.

Rezultatul acestui test este:



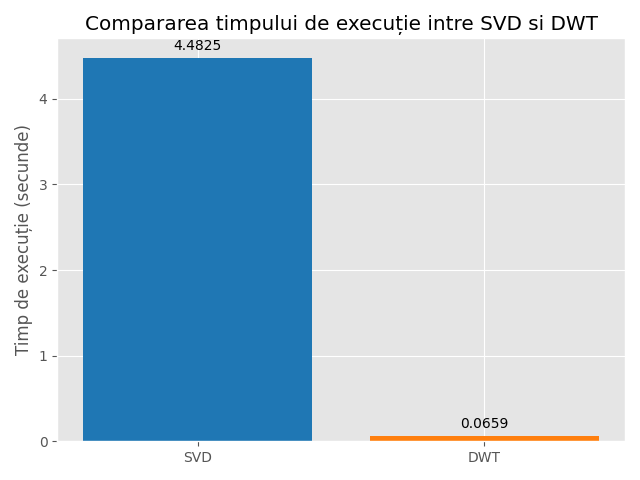
Pentru a evidenția importanța normalizării în contextul alterării, am executat un test în care am împărțit matricea la gradul de normalizare 5.



(dimensiune bloc = 75)

**6. Concluzii:**  
Pe baza datelor prezentate în graficul de mai jos, se poate trage concluzia că există o diferență notabilă între performanțele temporale ale algoritmilor de watermarking folosind SVD și DWT.

Această diferență semnificativă în performanța temporală poate fi atribuită eficienței algoritmilor în procesarea și manipularea datelor de imagine. DWT, fiind o metodă care descompune imaginea în frecvențe și detalii, pare să fie mai adaptată și optimizată pentru operațiunile rapide necesare în procesul de watermarking. În schimb, SVD, care necesită descompunerea matricilor complexe, este o operație mai intensivă din punct de vedere computațional și, prin urmare, mai lentă.



**7. Bibliografie:**

1. [***https://en.wikipedia.org/wiki/Singular\_value\_decomposition***](https://en.wikipedia.org/wiki/Singular_value_decomposition)
2. [***https://www.sciencedirect.com/topics/engineering/shannon-entropy***](https://www.sciencedirect.com/topics/engineering/shannon-entropy)
3. [***https://en.wikipedia.org/wiki/Entropy\_(information\_theory)***](https://en.wikipedia.org/wiki/Entropy_(information_theory))
4. [***https://en.wikipedia.org/wiki/RGB\_color\_model***](https://en.wikipedia.org/wiki/RGB_color_model)
5. [***https://en.wikipedia.org/wiki/Wavelet\_transform***](https://en.wikipedia.org/wiki/Wavelet_transform)
6. [***https://pywavelets.readthedocs.io/en/latest/***](https://pywavelets.readthedocs.io/en/latest/)