Tehnici DSP în Compresia Imaginilor: Implementarea unui Encoder JPEG

Student: Daniel Wagner
Program: Master TCSI, Anul 1
Curs: Cercetare
Data: Ianuarie 2026
Demo online: https://sem1-cercetare.danielwagner.ro
Cod sursă: https://github.com/danielw98/jpeg-encoder

Cuprins

1. Introducere

Ce este procesarea digitală a semnalelor?

Procesarea digitală a semnalelor (Digital Signal Processing - DSP) reprezintă domeniul care se ocupă cu analiza, modificarea și sinteza semnalelor reprezentate în formă discretă [4]. Spre deosebire de procesarea analogică, DSP oferă precizie, repetabilitate și flexibilitate în implementare, permițând algoritmi complecși care ar fi imposibil de realizat în domeniul analog [5, p. 1-3].

Un semnal poate fi definit ca orice mărime fizică care variază în timp, spațiu sau altă variabilă independentă. În contextul DSP, semnalele sunt eșantionate și cuantizate pentru a fi reprezentate numeric, permițând procesarea lor pe sisteme de calcul.

Relevanța DSP pentru compresia datelor

Compresia datelor este una dintre cele mai importante aplicații ale DSP [6, cap. 7]. Principiul fundamental care stă la baza compresiei este **redundanța** - informația care poate fi eliminată fără pierderi semnificative de calitate perceptuală.

DSP permite identificarea și exploatarea a două tipuri de redundanță:

- **Redundanța statistică** - pattern-uri repetitive în date care pot fi codate eficient
- **Redundanța perceptuală** - informație pe care sistemul vizual/auditiv uman nu o poate percepe

Transformările din domeniul frecvențelor (Fourier, DCT, wavelets) sunt instrumente fundamentale DSP care permit separarea informației esențiale de cea redundantă [4, cap. 13].

De la Fourier la JPEG

Parcursul conceptual de la analiza Fourier la standardul JPEG ilustrează evoluția tehnicilor DSP:

An	Dezvoltare	Semnificație
1822	Transformata Fourier	Analiză teoretică a frecvențelor
1965	FFT (Cooley-Tukey) [9]	Calcul eficient pe calculatoare
1974	DCT (Ahmed et al.) [7]	Optimizat pentru semnale finite
1992	JPEG (ITU-T T.81) [1]	Standard industrial

Transformata Fourier a demonstrat că orice semnal poate fi descompus în componente sinusoidale. **Transformata Discretă a Cosinusului (DCT)** a optimizat această idee pentru semnale finite, concentrând energia în mai puțini coeficienți [7]. **JPEG** a standardizat aplicarea DCT pentru compresia imaginilor, devenind cel mai utilizat format de imagine din lume [8].

Acest referat explorează fundamentele matematice ale acestor transformări, implementarea lor în algoritmul JPEG, și demonstrează conceptele printr-un encoder JPEG funcțional dezvoltat în C++.

2. Fundamente teoretice DSP

2.1 Transformata Fourier

Transformata Fourier reprezintă unul dintre cele mai importante instrumente matematice în procesarea semnalelor [5, cap. 2]. Descompune un semnal în componentele sale frecvențiale.

Transformata Fourier continuă:

$$X(f) = \int_{-\infty}^{\infty} x\,(t) \cdot e^{-j2\pi ft}\,dt$$

Transformata Fourier Discretă (DFT) pentru semnale digitale [5, cap. 8]:

$$X[k] = \sum_{n=0}^{N-1} x\,[n] \cdot e^{-j2\pi kn/N}$$

Complexitatea DFT este $O(N^2)$. Algoritmul **FFT** (Cooley-Tukey, 1965) [9] reduce la O(N log N).

Limitări pentru imagini: coeficienți complecși, discontinuități la margini, nu exploatează simetria [12].

2.2 Transformata Discretă a Cosinusului (DCT)

DCT-II (varianta JPEG) [7] transformă N eșantioane în N coeficienți frecvențiali:

$$X[k] = \alpha(k) \sum_{n=0}^{N-1} x\,[n] \cdot \cos\left[\frac{\pi(2n+1)k}{2N}\right]$$

Pentru JPEG, DCT 2D pe blocuri 8×8 [1, Anexa A]:

$$F(u,v) = \frac{1}{4}\alpha(u)\alpha(v) \sum_{x=0}^{7}\sum_{y=0}^{7} f\,(x,y)\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right]$$

Compactarea energiei: DCT concentrează 60-80% din energie în coeficientul DC [12, p. 150].

Avantaje față de DFT: coeficienți reali, compactare energetică superioară, continuitate la margini.

2.3 Noțiuni DSP conexe

Teorema Nyquist-Shannon [4, p. 31]: frecvența de eșantionare trebuie să fie cel puțin dublul frecvenței maxime: $f_s \geq 2 \cdot f_{max}$

Cuantizarea [6, p. 475] mapează valori continue la niveluri discrete. Este sursa principală de pierdere în compresia lossy.

Filtre digitale [5, cap. 5-7]: low-pass (blur), high-pass (edge detection), band-pass (extragere caracteristici).

3. Algoritmul JPEG și rolul DSP

Standardul JPEG (Joint Photographic Experts Group), definit în ITU-T T.81 [1], este cel mai utilizat format de compresie pentru imagini fotografice. Acest capitol detaliază fiecare etapă a pipeline-ului de codare conform specificației [11].

3.1 Preprocesare

Conversia spațiului de culoare: RGB la YCbCr

Imaginile sunt capturate în format RGB (Red, Green, Blue), dar pentru compresie eficientă se convertesc la **YCbCr** [3]:

- **Y** (Luminanță) - strălucirea percepută, corespunzătoare vederii monocromatice
- **Cb** (Crominanță albastră) - diferența față de componenta albastră
- **Cr** (Crominanță roșie) - diferența față de componenta roșie

Formula de conversie (ITU-R BT.601) [3]:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

Motivație: Sistemul vizual uman este mult mai sensibil la luminanță decât la crominanță [6, p. 442]. Aceasta permite subsampling agresiv pe canalele de culoare fără pierdere vizuală semnificativă.

Subsampling cromatic

JPEG reduce rezoluția canalelor Cb și Cr exploitând insensibilitatea ochiului la detaliile de culoare [8]:

Format	Raport Y:Cb:Cr	Descriere	Economie date
4:4:4	1:1:1	Fără subsampling	0%
4:2:2	2:1:1	Subsampling orizontal	33%
4:2:0	4:1:1	Subsampling orizontal și vertical	50%

Formatul **4:2:0** este cel mai frecvent utilizat, oferind un compromis excelent între calitate și compresie. În acest format, pentru fiecare 4 pixeli luminanță există câte un singur pixel Cb și Cr.

3.2 Transformarea DCT

Partiționarea în blocuri 8×8

Imaginea este divizată în blocuri de 8×8 pixeli [1, sec. 4.2]. Această dimensiune reprezintă un compromis optim între:

- **Eficiența compresiei** - blocuri mai mari ar concentra energia mai bine
- **Artefactele de blocking** - blocuri mai mari ar produce discontinuități mai vizibile
- **Complexitatea computațională** - 8×8 permite optimizări hardware eficiente

Valorile pixelilor sunt centrate la zero scăzând 128 (level shift), convertind intervalul [0, 255] în [-128, 127].

Forward DCT

Fiecare bloc 8×8 este transformat folosind DCT-II 2D [1, Anexa A]:

$$F(u,v) = \frac{1}{4}\alpha(u)\alpha(v)\sum_{x=0}^{7}\sum_{y=0}^{7} f\,(x,y)\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right]$$

unde $\alpha(0) = 1/\sqrt{2}$ și $\alpha(k) = 1$ pentru $k > 0$.

Rezultat: Matricea 8×8 de pixeli spațiali este transformată într-o matrice 8×8 de coeficienți frecvențiali. Coeficientul F(0,0) reprezintă valoarea medie (DC), iar ceilalți reprezintă variații de frecvență crescătoare (AC).

Cuantizarea

Coeficienții DCT sunt împărțiți la valorile dintr-o **matrice de cuantizare** și rotunjiți [1, sec. 4.3.1]:

$$F_q(u,v) = \text{round}\left(\frac{F(u,v)}{Q(u,v)}\right)$$

Matricile standard pentru luminanță și crominanță sunt definite în Anexa K a specificației [1]. Caracteristici importante:

- Valori mici în colțul stânga-sus (păstrează frecvențele joase)
- Valori mari în dreapta-jos (cuantizează agresiv frecvențele înalte)
- Scalare cu factorul de calitate Q (1-100)

Matricea de cuantizare pentru luminanță (Q=50) are valori de la 16 (poziția DC) până la 121 (frecvențe înalte). Prima linie conține valorile: 16, 11, 10, 16, 24, 40, 51, 61. Ultima linie: 72, 92, 95, 98, 112, 100, 103, 99.

Scanarea Zig-Zag

Coeficienții cuantizați sunt reordonați într-o secvență 1D folosind pattern-ul zig-zag [1, Fig. 5]. Acest pattern traversează matricea diagonal, grupând:

- Coeficienții de frecvență joasă (nenuli) la început
- Coeficienții de frecvență înaltă (adesea zero) la sfârșit

Această reordonare creează secvențe lungi de zerouri consecutive, ideale pentru codare RLE.

3.3 Codare entropică

Codarea coeficientului DC (DPCM)

Coeficientul DC variază lent între blocuri adiacente datorită corelației spațiale. Se codează **diferența** față de blocul anterior folosind DPCM (Differential Pulse Code Modulation) [1, sec. F.1.2.1]:

$$\Delta DC_i = DC_i - DC_{i-1}$$

Diferențele au în general valori mai mici decât valorile absolute, necesitând mai puțini biți pentru codare.

Codarea coeficienților AC (RLE)

Coeficienții AC sunt codați folosind **Run-Length Encoding** [1, sec. F.1.2.2]:

- Se numără zerourile consecutive (run length)
- Se codează perechea (RRRRSSSS, value) unde RRRR = run length, SSSS = category

Simboluri speciale:

- **EOB** (End of Block) - indică că restul coeficienților sunt zero
- **ZRL** (Zero Run Length) - reprezintă 16 zerouri consecutive

Codarea Huffman

Simbolurile RLE sunt codate folosind **coduri Huffman** cu lungime variabilă [1, Anexa K.3]. Principiul Huffman: simbolurile frecvente primesc coduri scurte, simbolurile rare primesc coduri lungi.

Categorie	Interval valori	Biți necesari
0	0	0
1	-1, 1	1
2	-3..-2, 2..3	2
3	-7..-4, 4..7	3
…	…	…
11	-2047..-1024, 1024..2047	11

Structura fișierului JPEG (JFIF)

Formatul JFIF (JPEG File Interchange Format) [2] definește structura completă a fișierului:

Marker	Cod hex	Descriere
SOI	FF D8	Start of Image
APP0	FF E0	JFIF application marker
DQT	FF DB	Define Quantization Table(s)
SOF0	FF C0	Start of Frame (baseline DCT)
DHT	FF C4	Define Huffman Table(s)
SOS	FF DA	Start of Scan (date comprimate)
EOI	FF D9	End of Image

Între SOS și EOI se află datele comprimate (entropy-coded data), reprezentând secvența de biți Huffman.

4. Alte aplicații DSP

Procesarea digitală a semnalelor se regăsește în aproape toate domeniile tehnologice moderne [4].

4.1 Compresie audio: MP3 și MDCT

Formatul **MP3** folosește **MDCT** (Modified Discrete Cosine Transform) [4, cap. 11], o variantă DCT optimizată pentru ferestre suprapuse. MP3 exploatează **mascarea auditivă**: sunete puternice maschează sunete slabe din frecvențe apropiate.

Aspect	JPEG	MP3
Transformare	DCT 2D (8×8)	MDCT 1D (576 samples)
Mascare	Vizuală	Auditivă
Compresie	10:1 - 20:1	10:1 - 12:1

4.2 Imagistică medicală

DSP este esențial în reconstrucția imaginilor medicale [6, cap. 5]:

- **RMN:** Transformata Fourier 2D/3D inversă reconstruiește imaginea din spațiul K
- **CT:** Filtered Back-Projection (FBP) combină proiecții din multiple unghiuri

- **Ecografie:** Beamforming digital focalizează fasciculul ultrasonic

4.3 Filtre și detecția muchiilor

Filtrele digitale sunt blocuri fundamentale [5, cap. 5-7]: low-pass (blur, denoising), high-pass (edge detection), gradient (Sobel, Prewitt).

Algoritmul **Canny** [10] combină: filtrare Gaussian, calcul gradient, non-maximum suppression, hysteresis thresholding.

4.4 Comunicații digitale

DSP în comunicații [4]: OFDM (WiFi, 4G/5G) folosește FFT/IFFT, filtre adaptive pentru egalizare, codare vocală (AMR, Opus).

5. Proiect demonstrativ: JPEG Encoder

Pentru a ilustra conceptele prezentate, am implementat un encoder JPEG complet în C++, cu o interfață web interactivă pentru vizualizare și analiză.

Demo online: https://sem1-cercetare.danielwagner.ro

Cod sursă: https://github.com/danielw98/jpeg-encoder

5.1 Arhitectura implementării

Proiectul **jpegdsp** este organizat modular, reflectând structura pipeline-ului JPEG prezentat în Capitolul 3:

Modul	Fișiere	Rol în pipeline
Core	Image.cpp, Block.cpp, ColorSpace.cpp	Încărcare imagine, partiționare blocuri, RGB→YCbCr
Transforms	DCT.cpp	Forward/Inverse DCT-II cu tabele cosinus precalculate
JPEG	Quantization.cpp, ZigZag.cpp, RLE.cpp, Huffman.cpp	Cuantizare, scanare, codare entropică
API	JPEGEncoder.cpp	Interfață simplificată pentru utilizare externă

Pipeline-ul complet (10 etape): Încărcare → RGB→YCbCr → Subsampling 4:2:0 → Partiționare 8×8 → Level shift → DCT → Cuantizare → Zig-Zag → DPCM/RLE → Huffman → JFIF output.

Fiecare etapă din acest pipeline este vizualizată în interfața web, în tab-ul **Pipeline**, unde utilizatorul poate explora detalii și teorie DSP pentru fiecare pas.

5.2 Interfața web

Interfața web oferă vizualizare interactivă a întregului proces de codare JPEG:

Stack tehnologic:

- **Backend:** Node.js + Express + TypeScript (port 3001)
- **Frontend:** React 18 + Vite + TypeScript + FontAwesome (port 3000)
- **Encoder:** C++17 CLI cu output JSON (30+ metrici)

Tab-uri disponibile:

- **Compare** - Comparație original vs comprimat cu slider calitate
- **Stats** - Statistici detaliate: blocuri, DCT, RLE, Huffman
- **Pipeline** - Vizualizare 10 etape cu teorie DSP interactivă
- **DCT** - Heatmap coeficienți, matrice cuantizare, energie
- **Chart** - Grafic interactiv calitate vs compresie

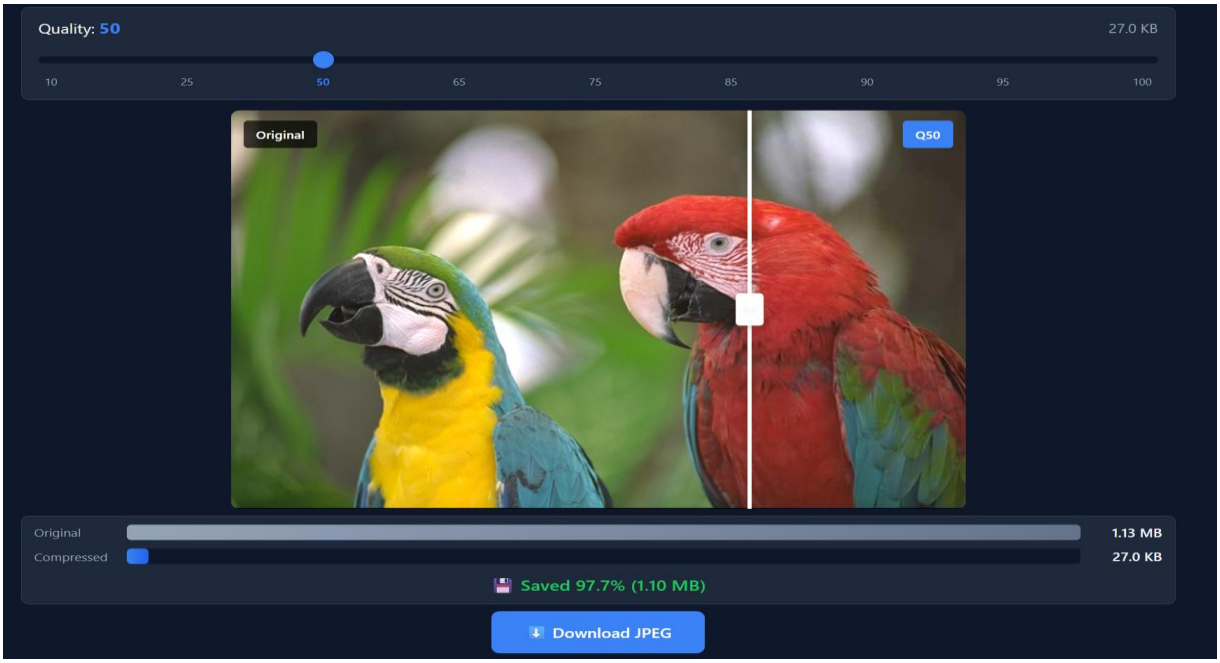

Figura 5.1: Tab-ul Compare cu slider pentru ajustarea calității

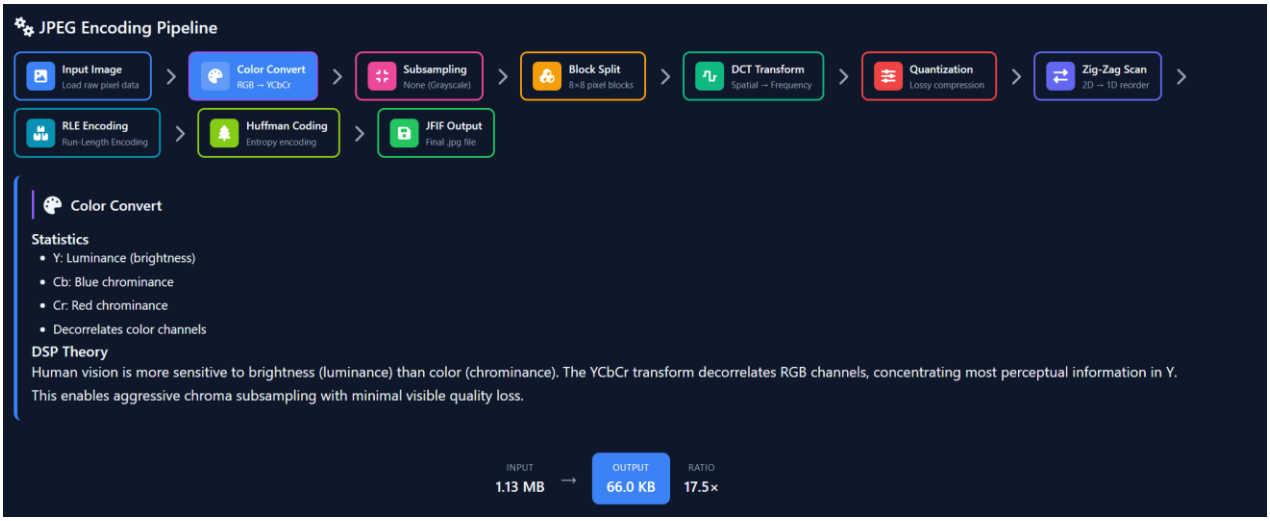

Figura 5.2: Tab-ul Pipeline cu cele 10 etape ale codării JPEG

Implementare tab Pipeline: Fiecare etapă afișează statistici calculate în timp real și explicații teoretice. De exemplu, etapa DCT arată distribuția energiei DC vs AC, iar etapa Huffman arată entropia înainte și după codare.

5.3 Rezultate experimentale

Testare pe imagini standard USC-SIPI și Kodak PhotoCD [16]:

Imagine	Original	Q=75	Q=50	Q=25
Baboon 512×512	768 KB	65 KB (11.8×)	40 KB (19.2×)	24 KB (32×)
Peppers 512×512	768 KB	48 KB (16×)	31 KB (24.8×)	19 KB (40.4×)
Kodim01 768×512	1.15 MB	78 KB (15.1×)	52 KB (22.6×)	33 KB (35.6×)

Distribuția energiei DCT:

Componentă	Energie	Descriere
DC (1 coeficient)	65-75%	Valoarea medie a blocului
AC joasă (frecvențe 1-3)	20-28%	Variații lente
AC medie (frecvențe 4-6)	4-8%	Detalii fine
AC înaltă (frecvențe 7)	1-3%	Zgomot, texture

Aceste rezultate validează că DCT concentrează energia în frecvențele joase, permițând cuantizarea agresivă a frecvențelor înalte.

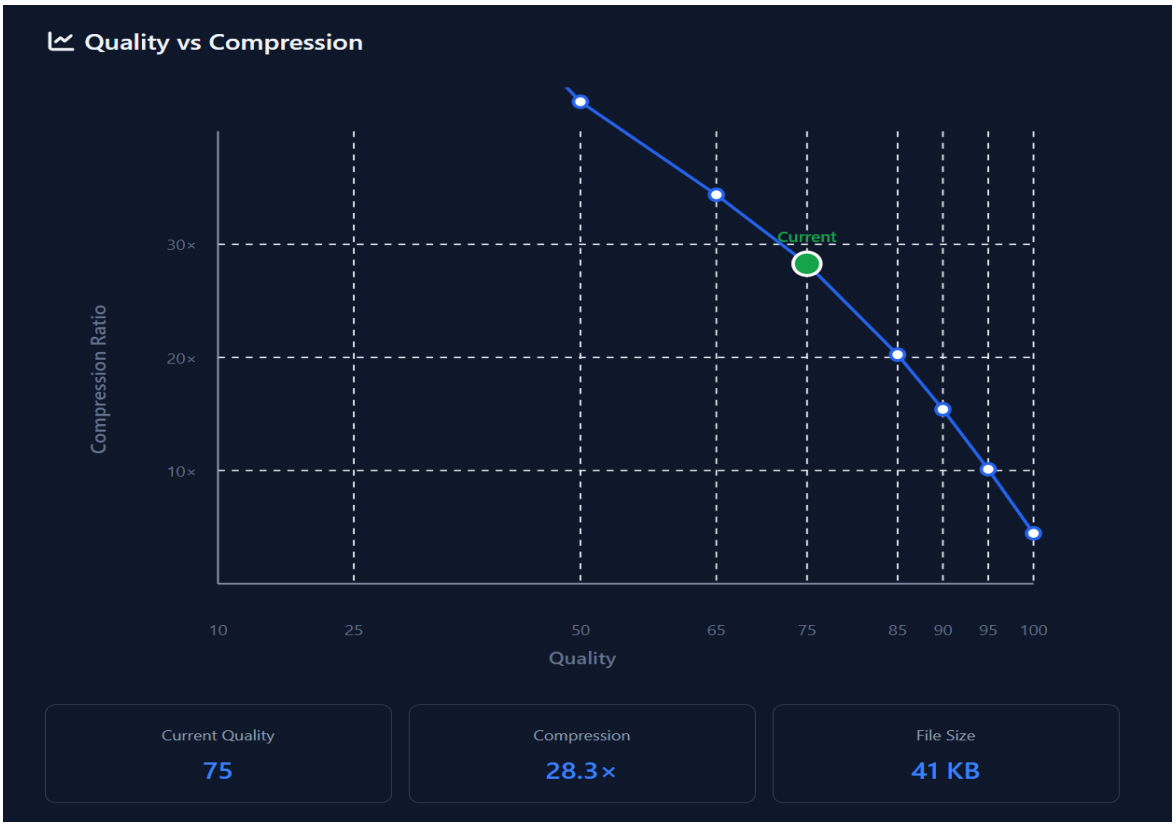

Figura 5.3: Grafic interactiv - relația dintre factorul de calitate și rata de compresie

5.4 Validare și complexitate proiect

Validare: Imaginile generate sunt compatibile cu vizualizatoare standard (Windows Photos, libjpeg-turbo [15]). Proiectul include 42 teste automatizate cu 100% succes.
Complexitate proiect: - ~4000 linii C++ (encoder core + tests + cli API) - ~2000 linii TypeScript (backend + frontend) - 30+ metrici de analiză în timp real - Cache sistem pentru precompute la 9 niveluri de calitate - Deploy Docker cu Nginx reverse proxy

6. Concluzii și direcții viitoare

6.1 Sinteza conceptelor

Implementarea encoder-ului JPEG a ilustrat modul în care tehnicile de procesare digitală a semnalelor permit compresia eficientă a imaginilor:
Transformata DCT [7] separă informația esențială de cea redundantă, concentrând energia în coeficienții de frecvență joasă. Această proprietate de compactare energetică este fundamentul compresiei lossy.
Cuantizarea adaptivă [1, Anexa K] exploatează modelul HVS (Human Visual System) al percepției umane. Sistemul vizual este mai sensibil la frecvențele joase și la luminanță, permițând cuantizare agresivă pe frecvențele înalte și pe crominanță.
Codarea Huffman [1, 11] optimizează reprezentarea statistică a simbolurilor. Alocarea codurilor scurte simbolurilor frecvente reduce semnificativ numărul de biți necesari.
Subsampling-ul cromatic exploatează sensibilitatea redusă a ochiului la detaliile de culoare, reducând volumul de date cu până la 50% fără degradare perceptibilă semnificativă.

6.2 Extensie: Prezentare interactivă Wavelet

Ca extensie a proiectului JPEG, am dezvoltat o **prezentare interactivă despre transformata wavelet**, disponibilă în repository-ul proiectului (prezentare_wavelet/).
Demo: https://dsp.danielwagner.ro/
Stack: FastAPI (Python) + React + PyWavelets
Prezentarea acoperă parcursul de la fundamentele Fourier până la JPEG2000:

- **Transformata Fourier** - Analiză spectrală, FFT interactiv
- **Filtre digitale** - Low-pass, high-pass, conexiunea cu wavelets
- **Convoluție** - Animații 1D/2D, kernel-uri (blur, sharpen, edge detection)
- **Wavelets** - Familii wavelet (Haar, Daubechies, Biorthogonal), scale/shift
- **Algoritmul Mallat** - Descompunere 2D multi-nivel (LL/LH/HL/HH)
- **Denoising** - Thresholding wavelet pentru eliminarea zgomotului
- **DCT vs Wavelet** - Comparație JPEG vs JPEG2000

Această extensie complementează proiectul principal, oferind o perspectivă asupra direcției în care a evoluat compresia imaginilor după standardul JPEG din 1992.

6.3 Direcții viitoare: Securitate și optimizare

Pentru dezvoltarea ulterioară a aplicației web, identificăm următoarele direcții prioritare:
Securitate API

- **Autentificare și autorizare** - Implementarea JWT tokens pentru acces controlat
- **Validare input** - Sanitizare riguroasă a fișierelor încărcate (verificare magic bytes, limite dimensiune)
- **CORS restrictiv** - Configurare whitelist pentru origini permise
- **HTTPS obligatoriu** - Criptare TLS pentru toate comunicațiile

Rate limiting și protecție DoS

- **Rate limiting per IP** - Limitarea numărului de cereri de encoding (CPU-intensive)
- **Queue management** - Sistem de coadă pentru procesări simultane
- **Timeout-uri** - Limite de timp pentru operații de encoding
- **Monitoring** - Alertare pentru pattern-uri de trafic anormale

Optimizare caching

- **Cache invalidation** - Strategii inteligente de invalidare (LRU, TTL)
- **CDN integration** - Distribuție geografică pentru imagini statice
- **Compression** - gzip/brotli pentru răspunsuri API
- **ETag headers** - Cache condiționat pentru resurse statice

Scalabilitate

- **Horizontal scaling** - Kubernetes sau Docker Swarm pentru mai multe instanțe
- **Load balancing** - Distribuție inteligentă a cererilor
- **Database caching** - Redis pentru cache-ul rezultatelor frecvente
- **Background workers** - Procesare asincronă pentru encoding-uri mari

6.4 Concluzie finală

JPEG rămâne, la peste 30 de ani de la standardizare [1], cel mai utilizat format de imagine din lume. Succesul său demonstrează eleganța cu care concepte matematice fundamentale (seria Fourier, DCT) pot fi aplicate pentru a rezolva probleme practice de compresie.
Proiectul demonstrativ dezvoltat ilustrează nu doar teoria din spatele algoritmului, ci și provocările practice ale implementării unei aplicații web moderne: arhitectură client-server, procesare în timp real, caching eficient, și considerații de securitate.

7. Bibliografie

Standarde

1 **ITU-T Recommendation T.81** (1992). *Information Technology – Digital Compression and Coding of Continuous-tone Still Images – Requirements and Guidelines*. International Telecommunication Union. ISO/IEC 10918-1.

2 **ITU-T Recommendation T.871** (2011). *Information Technology – Digital Compression and Coding of Continuous-tone Still Images: JPEG File Interchange Format (JFIF)*. International Telecommunication Union.

3 **ITU-R Recommendation BT.601** (2011). *Studio Encoding Parameters of Digital Television for Standard 4:3 and Wide-screen 16:9 Aspect Ratios*. International Telecommunication Union.

Cărți de referință DSP

4 **Lyons, R. G.** (2011). *Understanding Digital Signal Processing* (3rd ed.). Prentice Hall. ISBN: 978-0-13-702741-5.

5 **Oppenheim, A. V., & Schafer, R. W.** (2010). *Discrete-Time Signal Processing* (3rd ed.). Pearson. ISBN: 978-0-13-198842-2.

6 **Gonzalez, R. C., & Woods, R. E.** (2018). *Digital Image Processing* (4th ed.). Pearson. ISBN: 978-0-13-335672-4.

Articole fundamentale

7 **Ahmed, N., Natarajan, T., & Rao, K. R.** (1974). Discrete Cosine Transform. *IEEE Transactions on Computers*, C-23(1), 90-93. doi:10.1109/T-C.1974.223784

8 **Wallace, G. K.** (1991). The JPEG Still Picture Compression Standard. *Communications of the ACM*, 34(4), 30-44. doi:10.1145/103085.103089

9 **Cooley, J. W., & Tukey, J. W.** (1965). An Algorithm for the Machine Calculation of Complex Fourier Series. *Mathematics of Computation*, 19(90), 297-301.

10 **Canny, J.** (1986). A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6), 679-698.

Cărți specializate

11 **Pennebaker, W. B., & Mitchell, J. L.** (1993). *JPEG: Still Image Data Compression Standard*. Van Nostrand Reinhold. ISBN: 978-0-442-01272-4.

12 **Rao, K. R., & Yip, P.** (1990). *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Academic Press. ISBN: 978-0-12-580203-1.

13 **Mallat, S.** (2009). *A Wavelet Tour of Signal Processing* (3rd ed.). Academic Press. ISBN: 978-0-12-374370-1.

Resurse online

14 **JPEG.org** - Independent JPEG Group. https://jpeg.org/

15 **libjpeg-turbo** - Implementare optimizată JPEG. https://libjpeg-turbo.org/

16 **USC-SIPI Image Database** - Imagini de test standard. https://sipi.usc.edu/database/

Documentație software

17 **stb_image.h** - Sean Barrett. https://github.com/nothings/stb

18 **React Documentation** - Meta. https://react.dev/

19 **Express.js** - OpenJS Foundation. https://expressjs.com/