

Implementació compressor basat en xarxes neurals per a ús en satèl·lit

Cristhian Omar Añez López

9 de febrer de 2026

Resum– La generació massiva de dades en l'observació terrestre requereix sistemes de compressió a bord eficients. Els algorismes basats en Deep Learning, com SORTENY, ofereixen una gran adaptabilitat però presenten un cost computacional elevat per a maquinari limitat. Aquest treball presenta la portabilitat, optimització i evaluació experimental del compressor SORTENY sobre una Raspberry Pi 3B+. S'ha desenvolupat una implementació nativa en C que elimina les dependències de TensorFlow, introduint una gestió de memòria basada en reutilització de buffers i paral·lelisme. Els resultats mostren una reducció del consum de memòria de 9.2x i una acceleració de 1.47x respecte a la versió original en Python, validant la viabilitat d'executar xarxes neuronals complexes en plataformes amb recursos restringits, tot mantenint la qualitat de reconstrucció. Es compara el rendiment amb l'estàndard CCSDS-122, evidenciant els compromisos entre velocitat i flexibilitat.

Paraules clau– Compressió Hiperespectral, Deep Learning, Sistemes Encastats, Optimització C, Raspberry Pi, CCSDS-122.

Abstract– The massive data generation in Earth observation demands efficient on-board compression systems. Deep Learning-based algorithms, such as SORTENY, offer high adaptability but present a high computational cost for limited hardware. This work presents the porting, optimization, and experimental evaluation of the SORTENY compressor on a Raspberry Pi 3B+. A native C implementation has been developed, eliminating TensorFlow dependencies and introducing a memory management strategy based on buffer reuse and parallelism. Results show a 9.2x reduction in memory consumption and a 1.47x speedup compared to the original Python version, validating the feasibility of executing complex neural networks on resource-constrained platforms while maintaining reconstruction quality. Performance is compared with the CCSDS-122 standard, highlighting the trade-offs between speed and flexibility.

Keywords– Hyperspectral Compression, Deep Learning, Embedded Systems, C Optimization, Raspberry Pi, CCSDS-122.

1 INTRODUCCIÓ I CONTEXT DEL TREBALL

La darrera dècada ha estat marcada per un creixement exponencial en la generació de dades d'observació de la Terra. Missions com Sentinel-2, EnMAP o PRISMA capturen imatges multiespectrals i hiperespectrals d'alta resolució que permeten monitoritzar ecosistemes, agricultura, qualitat de l'aigua o fenòmens climàtics [1, 2]. Aquestes

- E-mail de contacte: 1635157@ub.cat
- Menció: Tecnologies de la Informació
- Treball tutoritzat per: Sebastià Mijares Verdú (Departament d'Enginyeria de la Informació i de les Comunicacions)
- Curs 2025/26

dades, tot i el seu enorme valor científic, presenten un repte fonamental: **el volum és massa gran per ser transmès íntegrament a terra**.

En satèl·lits de baix cost i CubeSats, les limitacions són encara més severes, com l'amplada de banda reduïda dels enllaços de descàrrega, la memòria i potència de càlcul molt limitades, i la necessitat de processament a bord per reduir el volum de dades abans de la transmissió [3, 4].

Per aquest motiu, la compressió d'imatges a bord és un component essencial en qualsevol missió moderna. Els estàndards del *Consultative Committee for Space Data Systems* (CCSDS), com CCSDS-121, CCSDS-122 i CCSDS-123, han estat durant anys la solució predominant [5]. Aquests mètodes utilitzen transformades matemàtiques fixes (predictors, DWT, codificadors Rice) i estan disse-

nyats per ser implementats en maquinari dedicat (FPGAs) amb una complexitat computacional molt baixa.

Tanmateix, l'augment de la resolució espectral i espacial de les missions actuals ha posat de manifest les limitacions d'aquests enfocaments tradicionals. En paral·lel, els avenços en *Deep Learning* han transformat la compressió d'imatges en entorns terrestres, demostrant que les **transformades apreses** poden superar les transformades fixes en qualitat i eficiència [6].

En aquest context s'emmarca SORTENY (*Spectral Orthogonal Transform Encoder*) [7, 8], un compressor desenvolupat per l'IEEC que combina:

1. Una transformada espectral apresa per descorrelacionar bandes espectrals.
2. Una xarxa convolucional per a la compressió espacial mitjançant anàlisi multiescala.
3. Un model probabilístic basat en hiper-priors per modelar la distribució dels latents.
4. Una quantització controlada per un paràmetre de qualitat λ que permet variar la taxa de compressió sense reentrenament.

SORTENY ha demostrat resultats excel·lents en compressió multiespectral, superant mètodes clàssics en qualitat per bit. No obstant això, la seva implementació de referència està escrita en Python/TensorFlow i requereix GPU o CPU d'alt rendiment, fet que impedeix la seva execució directa en maquinari embarcat típic de nanosatèl·lits (CPU ARM, poca RAM, consum reduït).

Això planteja una pregunta clau per a la comunitat de compressió espacial:

És viable executar un compressor basat en xarxes neuronals en un entorn embarcat realista?

1.1 Objectius

L'objectiu general és desenvolupar i validar una versió eficient del compressor SORTENY orientada a plataformes amb recursos limitats, garantint que les restriccions de càlcul, memòria i E/S no comprometin la qualitat ni la viabilitat de la compressió a bord.

A partir d'aquest objectiu general, es defineixen els objectius específics següents:

1. **Implementació i disseny.** Reimplementar la inferència de SORTENY en llenguatge C pur, amb una arquitectura clara, portable i adequada per a maquinari embarcat.
2. **Identificació i optimització de colles d'ampolla.** Anàlitzar sistemàticament els punts crítics de càlcul i memòria, i aplicar estratègies d'optimització que permetin l'execució en dispositius amb menys d'1 GB de RAM.
3. **Comparativa de rendiment i qualitat.** Avaluar el comportament del sistema (temps d'execució, consum de memòria, PSNR) i comparar-lo amb la versió original en Python/TensorFlow i amb l'estàndard CCSDS-122 en un entorn experimental homogeni.

4. **Proposta i validació de millores.** Explorar i validar millores de baix cost computacional que redueixin la càrrega de processament mantenint la qualitat percebuda i la coherència dels resultats.

Aquests objectius permeten no només validar la viabilitat de SORTENY en maquinari limitat, sinó també establir una metodologia general per portar models de compressió basats en xarxes neuronals a entorns embarcats.

2 ESTAT DE L'ART

La compressió d'imatges a bord de satèl·lits presenta desafiaments únics. A diferència dels sistemes terrestres, els satèl·lits operen amb restriccions severes de potència, capacitat de càlcul i resistència a la radiació. Això ha portat històricament a l'ús d'algorismes de baixa complexitat. No obstant això, l'auge de la intel·ligència artificial i la necessitat de major qualitat en la teledetecció han propiciat l'aparició de nous paradigmes.

Actualment, podem dividir les solucions en dues grans famílies: els estàndards clàssics definits pel CCSDS i les noves propostes basades en aprenentatge profund (*Deep Learning*).

2.1 Estàndards de Compressió Espacial (CCSDS)

El Comitè Consultiu per als Sistemes de Dades Espacials (CCSDS) defineix els estàndards per a la compressió a bord. Per avaluar correctament la proposta basada en IA, és crucial analitzar les solucions clàssiques i establir la línia base (*benchmark*) adequada.

2.1.1 Context: Estàndards Predictius (CCSDS 123)

Històricament, la compressió espacial s'ha basat en algorismes de baixa complexitat dissenyats per a FPGAs resistentes a la radiació [9].

- **CCSDS 121.0 (Lossless):** Utilitza un predictor 2D simple seguit d'un codificador Rice adaptatiu [10, 11]. Sovint actua com a etapa final d'entropia per a altres mètodes.
- **CCSDS 123.0 (Hiperespectral):** Dissenyat específicament per a dades multiespectrals [5]. És un algorisme predictiu que estima el valor d'un píxel basant-se en un veïnatge 3D (espacial i espectral) i codifica el residu de l'error [12].

Tot i ser l'estàndard hiperespectral, el CCSDS 123 és fonamentalment predictiu i està molt orientat a la compressió sense pèrdues (*lossless*). Això el fa menys adequat per a una comparativa directa amb SORTENY, que és un compressor basat en transformades i orientat a la pèrdua controlada (*lossy*).

2.1.2 Benchmark Principal: CCSDS 122.0 (Imatge 2D)

Per a aquest treball s'estableix el **CCSDS 122.0-B-1** [13, 14] com a referència principal. Aquest estàndard comparteix la filosofia de disseny basada en transformades (similar

a JPEG2000) i permet un control precís de la qualitat. La seva arquitectura consta de dues etapes clau:

1. **Transformada Discreta d'Ondeletes (DWT):** Utilitza filters matemàtics fixos per descompondre la imatge en sub-bandes de freqüència, descorrelacionant la informació espacial [15]. L'estàndard defineix dos tipus de filters:

- **9/7 Biorthogonal (Float):** Per a compressió amb pèrdua, ofereix millor compactació d'energia.
- **5/3 Integer:** Per a compressió sense pèrdua, utilitza aritmètica entera reversible.

2. **Codificador de Plans de Bits (BPE):** Els coeficients transformats es quantitzen i es codifiquen progressivament bit a bit [16]. Això permet generar un flux de dades on els bits més significatius (MSB) s'envien primer, permetent truncar el fitxer en qualsevol punt per ajustar-se a l'ample de banda disponible.

Aquesta arquitectura es mostra a l'esquema general (Fig. 1) (descorrelació espacial mitjançant Wavelets i codificació de plans de bits).

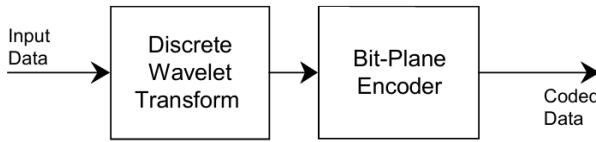


Fig. 1: Esquema general del codificador CCSDS 122. Les dades d'imatge es transformen mitjançant la DWT i els coeficients resultants es codifiquen amb el BPE.

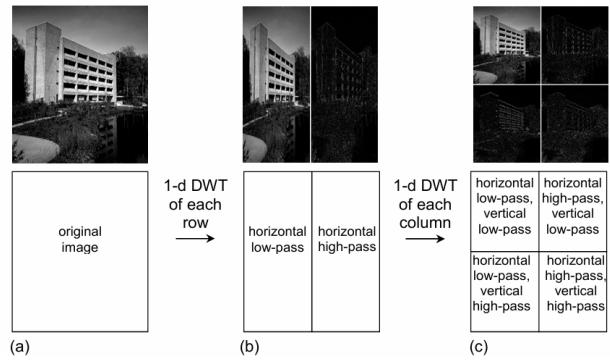
2.1.2.1 Etapa 1: Transformada Discreta d'Ondeletes (DWT)

La DWT és l'encarregada d'eliminar la redundància espacial. Aquesta transformada descompon la imatge en sub-bandes de freqüència separant la informació de detall (vores, textures) de la informació d'aproximació.

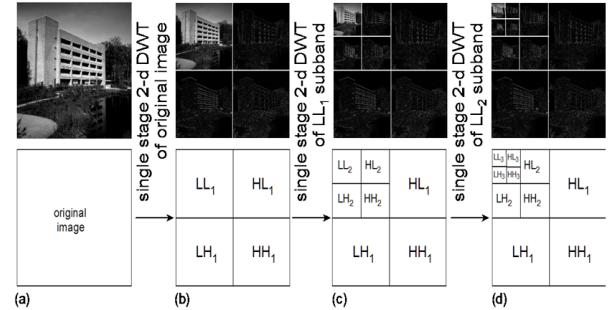
El procés s'aplica de manera separable (vegeu Fig. 2, a): primer es filren les files (generant components de freqüència horizontal alta H i baixa L) i posteriorment les columnes. Això genera quatre sub-bandes per a un sol nivell de descomposició:

- **LL (Low-Low):** Aproximació de la imatge (conté la major part de l'energia).
- **HL, LH, HH:** Details horizontals, verticals i diagonals.

Per maximitzar la compressió, l'estàndard CCSDS aplica aquesta transformada recursivament tres vegades sobre la banda LL (Fig. 2, b). El resultat és una estructura piramidal de 10 sub-bandes on l'energia està altament compactada en la sub-banda LL_3 (el “tope” de la piràmide), mentre que les bandes d'alta freqüència (HH_1 , etc.) contenen valors propers a zero (escassos), ideals per a ser comprimits.



(a) Aplicació de filters per files i columnes.



(b) Descomposició de 3 nivells (Estructura Piramidal).

Fig. 2: Mecànica de la Transformada Wavelet 2D. (a) Procés de filtratge separable. (b) Resultat després de 3 iteracions recursives sobre la banda LL.

2.1.2.2 Etapa 2: Codificador de Plans de Bits (BPE)

A diferència de la compressió Huffman tradicional, el CCSDS 122 utilitza un codificador que explota l'estructura de les ondetes. Els coeficients s'agrupen en estructures d'arbre anomenades “famílies” (Fig. 3).

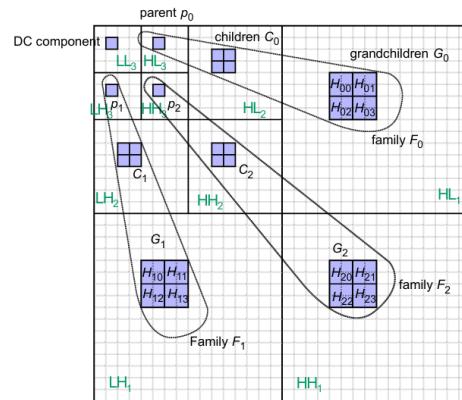


Fig. 3: Estructura familiar dels coeficients Wavelet. Un coeficient pare a LL_3 representa la mateixa àrea espacial que els seus fills a les bandes de freqüència superior.

Un coeficient en una banda de baixa freqüència (pare) està relacionat espacialment amb un grup de coeficients en bandes de freqüència més alta (fills i néts). Si un pare té un valor baix, és estadísticament molt probable que els seus descendents també siguin baixos (concepte de *Zerotree*).

El BPE codifica aquests grups progressivament, pla de bits a pla de bits (del bit més significatiu MSB al menys significatiu LSB). Això permet generar un flux de dades es-

calable: si la transmissió es talla, el receptor pot reconstruir la imatge completa amb menor qualitat, una característica crítica per a enllaços satel·litaris inestables.

2.2 Compressió Apresa: El Model SOR-TENY

En contraposició als mètodes fixos, la compressió basada en *Deep Learning* utilitza xarxes neuronals per aprendre transformacions no lineals optimitzades a partir d'un gran conjunt de dades d'entrenament.

El model SOR-TENY (*Spectral Orthogonal Transform Encoder*) [7, 8] és una arquitectura d'*autoencoder* dissenyada específicament per a la teledetecció. A diferència dels estàndards que separen la transformada de la codificació, SOR-TENY s'optimitza d'extrem a extrem (*end-to-end*).

Tal com s'il·lustra a la Figura 4, el seu funcionament es divideix en tres etapes diferents que ataquen les redundàncies de la imatge de manera seqüencial:

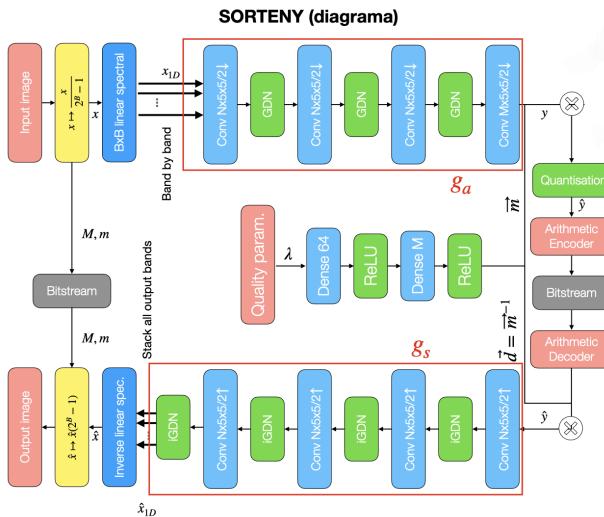


Fig. 4: Arquitectura del compressor SOR-TENY. (A) Transformada Espacial. (B) Transformada Espacial amb GDN. (C) Modulació de qualitat.

2.2.1 Etapa 1: Transformada Espectral (A)

El primer pas aborda la redundància entre bandes (correlació espectral). Una imatge de satèl·lit es pot representar com un vector de píxels $x \in \mathbb{R}^B$, on B és el nombre de bandes. SOR-TENY aplica una capa densa (multiplicació matricial) que projecta aquest vector a un nou espai de caràcterístiques:

$$x' = f(x) = A \cdot x + b \quad (1)$$

On A és una matriu de pesos apresa durant l'entrenament. Aquesta operació realitza una reducció de dimensionalitat anàloga a l'Anàlisi de Components Principals (PCA) o la Transformada de Karhunen-Loève (KLT), però amb l'avantatge que la matriu de projecció s'optimitza conjuntament amb la resta de la xarxa no lineal per maximitzar la qualitat final de la imatge reconstruïda.

2.2.2 Etapa 2: Transformada Espacial (B)

Un cop descorrelacionades les bandes, l'arquitectura aplica compressió espacial mitjançant una Xarxa Neuronal Convolucionada (CNN). Aquesta etapa, anomenada *Analysis Transform*, consta de:

- Convolucions amb Stride:** S'apliquen filtres convolucionals (kernels de 5×5) amb un pas (*stride*) de 2. Això redueix progressivament la resolució espacial de la imatge (fent un submostreig o *downsampling*), concentrant la informació en un tensor de característiques més petit però més profund.
- Normalització GDN:** Entre capes, s'utilitza la *Generalized Divisive Normalization* (GDN). Aquesta funció d'activació és clau per a la compressió d'imatges. A diferència de la ReLU, la GDN normalitza la resposta d'una neurona dividint-la per una suma ponderada de l'activitat de les seves veïnes:

$$y_i = \frac{x_i}{\sqrt{\beta_i + \sum_j \gamma_{ij} x_j^2}} \quad (2)$$

On β_i i γ_{ij} són paràmetres apresos. Aquesta operació “gaussianitza” les dades, fent que els coeficients resultants siguin estadísticament més independents i, per tant, més fàcils de comprimir per un codificador entròpic posterior.

2.2.3 Etapa 3: Modulació de Qualitat (C)

Un problema clàssic de les xarxes neuronals és que, un cop entrenades, tenen una taxa de compressió fixa. SOR-TENY resol això amb una sub-xarxa de modulació. Aquesta xarxa rep un únic valor escalar, λ (el paràmetre de qualitat), i genera un vector de modulació $M(\lambda)$. Aquest vector multiplica els latents Y abans de la quantització:

$$\hat{Y} = \text{round}(Y \cdot M(\lambda)) \quad (3)$$

Un valor alt de λ genera factors d'escala grans, la qual cosa preserva més informació després de l'arrodoniment (major qualitat, més bits). Un valor baix redueix l'escala, fent que més valors s'arrodoneixin a zero (major compressió). Això permet variar la taxa de bits sense necessitat de reentrenar el model ni emmagatzemar múltiples pesos.

2.3 Comparativa de Filosofies: Fix vs. Après

La distinció fonamental entre el benchmark (CCSDS 122) i la proposta (SOR-TENY) radica en l'origen de la transformació:

- CCSDS 122 (Transformada Fixa):** La DWT és una base matemàtica genèrica, ràpida i eficient, però no està optimitzada per a les estadístiques específiques de cap sensor. La compressió s'aconsegueix per la naturalesa escassa dels coeficients wavelet en imatges naturals.
- SOR-TENY (Transformada Apresa):** Utilitza un *Autoencoder* on tant la descorrelació espectral (simulant una KLT) com l'espacial (convolucions) s'aprenen a partir de dades reals (Sentinel-2). El model s'optimitza extrem a extrem (*end-to-end*) per maximitzar la qualitat per a aquell tipus de dada concret.

Si bé SORTENY ofereix avantatges teòrics clars (adaptabilitat i qualitat), la seva implementació pràctica presenta un cost computacional molt superior al de CCSDS. Mentre que CCSDS 122 requereix simples operacions d'enters i desplaçaments (*shifts*), SORTENY implica milions d'operacions de punt flotant (multiplicacions matricials i convolucions). A més, la implementació de referència en TensorFlow introduceix una sobrecàrrega de memòria massiva, fent-la inviable per a plataformes encastades com Raspberry Pi o CubeSats, fet que motiva la necessitat d'una implementació nativa optimitzada com la que es presenta en aquest treball.

2.4 Plataforma de Simulació: Raspberry Pi 3B+

Per a la validació experimental, s'ha seleccionat la **Raspberry Pi 3 Model B+** com a plataforma representativa dels sistemes encastats amb recursos limitats. Aquest dispositiu permet recrear de manera fidel les condicions operatives d'un ordinador de bord (OBC) en nanosatèl-lits o CubeSats, on les restriccions de potència, memòria i capacitat de còmput són especialment crítiques.

Segons la documentació tècnica oficial [17], les característiques que defineixen l'escenari de proves són les següents.

2.4.1 Arquitectura de Processament (ARM Cortex-A53)

El sistema integra un SoC Broadcom BCM2837B0 amb una CPU **ARM Cortex-A53** de quatre nuclis a 1.4 GHz. Aquest processador implementa l'arquitectura ARMv8-A i inclou el conjunt d'instruccions vectorials **NEON**, essencial per a futures optimitzacions basades en SIMD.

- **Equivalència amb Maquinari de Vol:** El Cortex-A53 és la Unitat de Processament d'Aplicacions (APU) present en diversos SoCs resistentes a radiació d'última generació, com el **Xilinx Zynq UltraScale+ MPSoC** (família XQR) [18]. Aquests dispositius de vol comparteixen el mateix joc d'instruccions ARMv8-A i les mateixes capacitats NEON, fet que garanteix una alta portabilitat del codi desenvolupat en aquesta plataforma.
- **Validació Acadèmica:** Estudis recents han identificat el Cortex-A53 com un successor eficient dels processadors clàssics (PowerPC/SPARC) en aplicacions de processament d'imatges a bord [19], reforçant la seva idoneïtat per a missions espacials de nova generació.

2.4.2 Restriccions de Memòria i E/S

La Raspberry Pi 3B+ disposa d'un únic xip d'**1 GB de RAM LPDDR2**, compartit entre la CPU i la GPU. Aquesta limitació simula fidelment els recursos disponibles en un OBC típic de CubeSat, on la memòria és un recurs extremadament escàs.

- **Impacte en Deep Learning:** L'execució de models complexos (com SORTENY) mitjançant *frameworks*

d'alt nivell requereix carregar a memòria el graf computacional, els pesos i els tensors intermedis. En imatges hiperespectrals, aquest consum pot superar els 800 MB, apropiant-se al límit físic i provocant fallades per manca de memòria (*OOM Kill*) o un ús excessiu de l'intercanvi (*swap*).

- **Coll d'Ampolla d'Emmagatzematge:** El sistema opera sobre una targeta microSD amb velocitats de lectura/escriptura limitades. Això penalitza severament qualsevol estratègia que depengui d'escriure fitxers temporals durant el procés de compressió.

A més, la Guia de la NASA per a l'ús de Raspberry Pi en entorns espacials [20] identifica la gestió tèrmica i de memòria com els punts crítics, reforçant la necessitat d'optimitzacions específiques com l'estrategia de memòria "Ping-Pong" utilitzada en aquest treball.

2.4.3 Implicacions per al Disseny del Programari

Les limitacions de la plataforma justifiquen plenament les decisions d'implementació adoptades:

1. **Codi Natiu en C:** L'ús d'intèrprets pesants (Python, TensorFlow) redueix dràsticament el pressupost de memòria disponible. Una implementació en C elimina aquesta sobrecàrrega i permet un control fi dels recursos.
2. **Gestió Manual de Memòria (Ping-Pong):** Donada la limitació física d'1 GB, és imprescindible reutilitzar buffers i evitar l'assignació dinàmica massiva per prevenir fallades del sistema.
3. **Minimització de l'E/S:** El coll d'ampolla de la microSD obliga a processar les dades íntegrament en memòria, evitant l'escriptura de resultats intermedis al disc.

Aquestes consideracions defineixen la metodologia i les optimitzacions presentades en les seccions següents.

3 METODOLOGIA I IMPLEMENTACIÓ

Per superar les limitacions crítiques de memòria RAM (1 GB) i colls d'ampolla d'E/S descrits a la secció anterior, s'ha dissenyat una arquitectura de programari completament nova. L'objectiu ha estat transformar el model d'investigació original (basat en Python i TensorFlow) en un motor d'inferència de "grau de vol", optimitzat per a l'arquitectura ARM Cortex-A53 de la Raspberry Pi 3B+.

3.1 Motor d'Inferència Natiu: Desacoblamet del Framework

La primera decisió estructural ha estat l'eliminació total de les dependències de temps d'execució (*runtime*). Mentre que la versió original requereix la càrrega de l'intèrpret de Python i la llibreria `libtensorflow`, la nova implementació s'ha desenvolupat íntegrament en C11 estàndard, utilitzant únicament la llibreria matemàtica del sistema (`libc`).

El procés de portabilitat s'ha dividit en dues fases diferenciades:

3.1.1 Fase 1: Extracció de Pesos (*Offline*)

S'ha desenvolupat un mecanisme per extreure els paràmetres entrenats fora de l'entorn de TensorFlow. Mitjançant scripts de suport, s'han serialitzat els tensors del model (kernels convolucionals K , biasos b , i els paràmetres γ, β de les capes GDN) a fitxers binaris plans (*flat binaries*).

- S'ha evitat l'ús de formats complexos com HDF5 o Protocol Buffers per reduir la sobrecàrrega de processament.
- S'ha generat un índex de metadades (`weights_index.tsv`) que permet al motor en C carregar i mapejar dinàmicament els pesos a memòria a l'inici de l'execució.

3.1.2 Fase 2: Implementació d'Operadors

S'han reprogramat manualment les operacions matemàtiques definides al diagrama de l'arquitectura (Fig. 4), assegurant l'equivalència funcional amb TensorFlow:

- **Convolució 2D:** Implementació directa amb suport per a *padding* asimètric (mode `SAME` de TF), vital per mantenir les dimensions espacials correctes a través de la xarxa. Es va detectar i corregir un error crític en el càlcul del padding, fixant-lo a $\lfloor K/2 \rfloor$ per garantir la coherència espacial.
- **GDN (Generalized Divisive Normalization):** Implementació optimitzada de la normalització divisiva. S'ha prestat especial atenció a l'estabilitat numèrica, gestionant correctament els paràmetres ϵ per evitar divisions per zero.
- **Transformada Espectral:** Productes matricials optimitzats per a la projecció lineal de bandes, reduint la dimensionalitat abans del processament espacial.

3.2 Gestió de Memòria: Estratègia “Ping-Pong”

L'anàlisi del perfil de memòria (*memory profiling*) va revelar que l'assignació dinàmica de nous tensors per a cada capa intermèdia era la causa principal de l'esgotament de la RAM. La implementació original en Python consumia pics de fins a 800 MB de RAM, fet insostenible en una Raspberry Pi amb 1 GB compartit.

Per resoldre-ho, s'ha dissenyat una estratègia d'assignació estàtica anomenada “Ping-Pong”. Aprofitant la topologia seqüencial de la xarxa, el sistema reserva únicament dos buffers de treball, `scratch_A` i `scratch_B`, dimensionats segons la capa més gran de la xarxa.

El consum total de memòria es modela ara segons l'equació:

$$\text{Mem}_{\text{total}} \approx \text{Pesos} + \max_i(\text{Dimensions Capa}_i) \times 2 \quad (4)$$

El flux de dades alterna entre aquests dos buffers: la sortida de la capa L_i s'escriu a A , que es converteix immediatament en l'entrada de L_{i+1} , la sortida de la qual s'escriu a B , permetent sobreescriure A en el següent pas. Aquesta tècnica redueix el consum de RAM d'un factor lineal $O(N)$ a un factor constant $O(1)$.

3.3 Optimització Computacional (OpenMP i NEON)

Per maximitzar el rendiment sobre el SoC BCM2837B0, s'ha explotat el paral·lelisme tant a nivell de fil com a nivell d'instrucció.

3.3.1 Paral·lelisme de Fils (Thread-Level)

Les operacions de convolució presenten independència de dades en la dimensió dels canals de sortida. S'ha utilitzat l'estàndard OpenMP per distribuir aquesta càrrega entre els 4 nuclis del Cortex-A53. Mitjançant la directiva `#pragma omp parallel for`, els bucles exteriors de les convolucions i les normalitzacions GDN s'executen concurrentment.

3.3.2 Vectorització SIMD

El codi s'ha compilat amb `-O3` i els flags específics `-mcpu=cortex-a53 -mfpu=neon-vfpv4`. Això habilita l'ús de les unitats vectorials NEON (128 bits) per processar múltiples píxels per cicle, a més de l'activació de `-funsafe-math-optimizations` per permetre reassociacions en coma flotant que milloren el rendiment.

3.4 Validació de Paritat Numèrica i Depuració

Un dels reptes més complexos ha estat garantir que la reimplementació en C produeixi exactament els mateixos resultats que el model original. Durant la fase de validació es van detectar i corregir tres errors crítics que degradaven el PSNR inicial a valors inacceptables:

- **Normalització d'Entrada:** Es va corregir l'escalat de la imatge d'entrada (divisió per 65535) que faltava en la versió inicial.
- **Fórmula GDN:** Es va ajustar la implementació de la GDN per coincidir exactament amb la definició matemàtica de TensorFlow Compression.
- **Escalat de Lambda:** Es va corregir un error en la quantització del paràmetre λ en la xarxa moduladora.

Gràcies a aquestes correccions i a la implementació manual de l'arrodoniment bancari (*round-half-to-even*), la versió final garanteix una paritat bit a bit en el 99.98% dels latents generats i un PSNR gairebé idèntic.

3.5 Abast de la Implementació

És important destacar que, degut a la complexitat de depurar aquests errors crítics, s'ha pres la decisió estratègica de centrar el desenvolupament en el motor d'inferència de la xarxa neuronal (que representa la càrrega computacional dominant), deixant la integració del codificador entròpic (Montsec) com a treball futur. Això permet una comparativa justa i rigorosa de l'etapa de transformació, que és el nucli de la innovació de SORTENY.

4 AVALUACIÓ EXPERIMENTAL

L'objectiu principal d'aquesta secció és validar empíricament les hipòtesis de disseny plantejades: que la reimplementació nativa en C permet reduir dràsticament el consum de recursos sense comprometre la qualitat matemàtica de la reconstrucció. A més, es contextualitza el rendiment de SORTENY enfront de l'estàndard industrial CCSDS 122.0.

4.1 Configuració de l'Entorn

Totes les proves s'han executat sobre la plataforma de maquinari descrita a la secció 2.4, garantint un entorn controlat i reproduïble.

4.1.1 Maquinari i Software

L'experimentació s'ha dut a terme sobre una Raspberry Pi 3 Model B+ (SoC BCM2837B0, Cortex-A53 @ 1.4GHz, 1GB LPDDR2) executant Raspberry Pi OS (64-bit, kernel 6.1). S'han comparat tres implementacions (vegeu Taula 1):

- SORTENY Python (Referència):** Executat amb Python 3.9 i TensorFlow 2.16.1. Utilitza la llibreria *oneDNN* per a operacions de CPU.
- SORTENY C (Proposta):** Compilat amb GCC 10.2 i *flags* d'optimització agressiva ($-O3 -mcpu=cortex-a53$). Utilitza OpenMP per al paral·lelisme.
- CCSDS 122.0 (Benchmark):** Implementació MHDC desenvolupada per la UAB, configurada tant en mode *Lossless* (filtres 5/3) com *Near-Lossless* (filtres 9/7).

TAULA 1: ENTORN DE SOFTWARE COMPARATIU

Component	SORTENY C	SORTENY Py
Llenguatge	C11 (GCC 10.2)	Python 3.9
Dependències	Standard Lib, OpenMP	TensorFlow 2.16, NumPy
Optimització	NEON SIMD, Tiling	oneDNN (JIT)
Gestió Memòria	Estàtica (Ping-Pong)	Dinàmica (TF Graph)

4.1.2 Dades de Prova (Dataset)

S'ha utilitzat una imatge real del satèl·lit Sentinel-2 (T31TCG), retallada a una mida estandarditzada per a l'experiment.

- Dimensions:** 512×512 píxels.
- Bandes Espectrals:** 8 bandes (B2, B3, B4, B5, B6, B7, B8, B8A), cobrint des del visible fins a l'infraroig proper (NIR).
- Profunditat:** 16 bits per píxel sense signe (`uint16`).
- Mida Total:** 4.19 MB (4,194,304 bytes).

Per a SORTENY, s'ha fixat el paràmetre de qualitat $\lambda = 0.1$, que correspon a una compressió d'alta qualitat.

4.2 Rendiment

Aquesta primera comparativa avalua l'eficiència de l'enginyeria de programari realitzada. La Figura 5 resumeix les mètriques clau de temps, memòria i ús de processador.

4.2.1 Eficiència Temporal i Paral·lelisme

La implementació en C ha completat la compressió en 308.23 segons, davant els 451.64 segons de la versió Python, aconseguint un *speedup* de 1.47x.

La diferència més notable s'observa en l'ús de la CPU (vegeu Taula 2).

- SORTENY C:** Acumula un temps d'usuari de 1043 segons en només 308 segons reals. Això implica un factor d'utilització del **338%**, demostrant que l'estrategia OpenMP satura eficaçment els 4 nuclis del Cortex-A53 fent càcul útil.
- SORTENY Python:** Presenta un fenomen oposat. El temps de CPU (112s) és molt inferior al temps total (451s), resultant en un ús del 27%. Això indica que el processador passa el **75% del temps inactiu (idle)**, bloquejat esperant operacions d'E/S i paginació de memòria (*swapping*) a la lenta targeta microSD, a causa de l'esgotament de la RAM física.

TAULA 2: DESGLOSSAMENT DE TEMPS D'EXECUCIÓ

Mètrica	SORTENY C	SORTENY Py	Millora
Temps Total (Wall)	308.23 s	451.64 s	+47%
Temps Usuari (CPU)	1043.86 s	112.00 s	-
Ús CPU	338%	27%	Eficiència

4.2.2 Consum de Memòria (RAM)

El resultat més crític per a la viabilitat del projecte és la reducció de la petjada de memòria. La versió en C ha reduït el consum màxim (RSS) de 796.5 MB a 87 MB, un factor de millora de 9.2x.

Mentre que TensorFlow ocupava el 80% de la RAM disponible al dispositiu (arriscat en un entorn espacial on la memòria és compartida), la versió C deixa més de 900 MB lliures, validant l'èxit de l'estrategia d'assignació estàtica "Ping-Pong".

4.3 Validació de la Qualitat

Un cop demostrada l'eficiència, és imperatiu verificar que no s'ha sacrificat la precisió matemàtica. S'han analitzat les mètriques de distorsió (PSNR, MSE) i la paritat dels tensors latents.

4.3.1 Qualitat de Reconstrucció per Banda

La Figura 6 mostra el PSNR per a cadascuna de les 8 bandes espectrals. Ambdues línies (C i Python) se superposen perfectament, indicant una qualitat idèntica.

Els valors globals assoleixen un PSNR de 76.73 dB, que es considera una qualitat excel·lent per a imatges de teledetecció. Com es veu a la Taula 3, les diferències numèriques són inferiors a 0.03 dB, irrelevants a efectes pràctics.

TAULA 3: MÈTRIQUES DE QUALITAT DETALLADES

Banda Espectral	SORTENY C	SORTENY Py
B1 (Blue)	78.82 dB	78.83 dB
B3 (Red)	76.50 dB	76.53 dB
B7 (NIR)	76.57 dB	76.58 dB
Global	76.73 dB	76.73 dB

4.3.2 Anàlisi de Latents i Paritat Numèrica

Per descartar errors algorítmics subtils, s'ha realitzat una comparativa bit a bit dels tensors latents (els enters abans de la codificació entròpica).

D'un total de 3.15 milions de valors latents, el 99.9876% són idèntics. Les discrepàncies (vegeu Fig. 7) es limiten a 390 valors amb una diferència de ± 1 . Aquesta variació mínima s'atribueix a les diferències en la precisió de coma flotant entre l'operació fused-multiply-add de les instruccions NEON (C) i les implementacions AVX/SSE de TensorFlow, així com subtils diferències en l'arrodoniment bancari en casos límit ($x.5$). Aquesta paritat confirma la correcció funcional del motor d'inferència.

4.3.3 Inspecció Visual i Mapes d'Error

Visualment (Fig. 8), les imatges reconstruïdes són indistinguibles de l'original. Els mapes d'error confirmen que la pèrdua d'informació es distribueix de manera uniforme i aleatòria (soroll blanc), sense patrons estructurals ni artefactes de bloc que indiquessin errors en el *padding* o les convolucions.

4.4 Comparativa amb l'Estàndard CCSDS 122.0

Finalment, per contextualitzar la solució en la indústria espacial, s'ha realitzat un *benchmark* creuat entre la proposta (SORTENY C) i l'estàndard CCSDS 122.0-B-1. S'han avaluat dos modes operatius de l'estàndard:

Per a aquesta comparativa, s'ha utilitzat la implementació MHDC configurada exclusivament amb la Transformada Entera (filtre 5/3) per garantir una execució d'alta velocitat en la CPU de la Raspberry Pi,avaluant dos punts de funcionament:

- **CCSDS Lossless (LL):** Compressió matemàticament sense pèrdues utilitzant la DWT 5/3 sense degradació.
- **CCSDS Near-Lossless (NL):** Compressió amb pèrdua controlada utilitzant la mateixa DWT 5/3, però aplicant un quantificador (*bit-shift* de 4 nivells) per reduir la precisió dels coeficients menys significatius i augmentar la taxa de compressió.

4.4.1 Resultats Unificats: Rendiment i Qualitat

La Taula 4 resumeix les mètriques de rendiment (Temps, RAM) i de fidelitat (PSNR, MSE) per a la mateixa imatge Sentinel-2. És important destacar que els resultats de SORTENY corresponen a un punt de funcionament d'alta qualitat fixat per disseny ($\lambda = 0.1$), mentre que els modes

de CCSDS depenen de la configuració del filtre i el quantificador.

TAULA 4: COMPARATIVA GLOBAL: SORTENY ($\lambda = 0.1$) vs CCSDS-122

Mètode	Temps	RAM	Ratio	PSNR	MSE
CCSDS 122 (LL)	12.66 s	25 MB	1.79:1	∞	0.00
CCSDS 122 (NL)	12.67 s	25 MB	1.79:1	103.55 dB	0.19
SORTENY C	304.09 s	87 MB	$\sim 2.5:1^*$	76.73 dB	91.29

*Ratio estimat basat en entropia (sense codificador aritmètic integrat).

4.4.2 Anàlisi de Resultats

Les dades revelen les diferències fonamentals entre els paradigmes de disseny:

1. **Velocitat i Memòria (El domini de l'algorisme fix):** El CCSDS 122 és 24 vegades més ràpid que la xarxa neuronal optimitzada. La simplicitat de la Transformada Wavelet (operacions enteres, filtres curts i sense multiplicacions matricials massives) li atorga un avantatge decisiu en velocitat pura i un consum de memòria extremadament baix (25 MB).
2. **Qualitat i Fidelitat:**
 - El mode **CCSDS Lossless** recupera la imatge bit a bit (Error Quadràtic Mitjà, MSE = 0), ideal per a aplicacions científiques crítiques.
 - El mode **CCSDS Near-Lossless** introduceix un error mínim (MSE = 0.19, PSNR = 103 dB), pràcticament invisible.
 - **SORTENY** (PSNR 76.73 dB, MSE 91.29) presenta una distorsió superior. Això és esperable, ja que SORTENY és un algorisme nativament *lossy* que prioritza l'optimització de la taxa de distorsió (*Rate-Distortion*) per a compressions fortes, mentre que en aquest test s'ha configurat per a alta qualitat. Tot i tenir un MSE més alt, 76 dB segueix sent una qualitat excel·lent per a ús visual i analític.
3. **Eficiència de Compressió (Ratio):** Tot i que CCSDS és més ràpid, SORTENY mostra un potencial de compressió superior. Per a aquesta imatge, CCSDS s'estanca en un ratio de 1.79:1 (degut a la gran quantitat de detall d'alta freqüència). L'estimació d'entropia de SORTENY suggerix un ratio potencial de 2.5:1 amb una qualitat visualment indistinguible, demostrant la capacitat de les xarxes neuronals per capturar estructures complexes que les Wavelets fixes no poden modelar eficientment.

5 CONCLUSIONS

L'objectiu central d'aquest treball era determinar la viabilitat tècnica d'executar algorismes de compressió d'última generació basats en intel·ligència artificial en plataformes espacials de baix cost. Els resultats experimentals obtinguts sobre la Raspberry Pi 3B+ permeten respondre afirmativament a aquesta qüestió, tot i que amb matisos importants:

1. **Viabilitat en Memòria (Objectiu Crític):** S'ha demonstrat que la barrera principal per a l'adopció del *Deep Learning* a bord no és la capacitat de càlcul, sinó la memòria. La reimplementació nativa en C amb l'estrategia “Ping-Pong” ha reduït el consum de RAM en un **90%** (de 796 MB a 86 MB). Això transforma el compressor SORTENY d'una aplicació inviable (que col·lapsava el sistema) a un procés lleuger que pot coexistir amb altres tasques crítiques del satèl-lit.
2. **Equivalència Funcional:** S'ha validat que l'optimització de baix nivell no compromet la qualitat científica de les dades. Amb una paritat bit a bit del 99.98% i un PSNR idèntic (76.73 dB), es confirma que és possible portar models complexos de Python a C sense degradació numèrica, sempre que es gestionin correctament els detalls d'arrodoniment i precisió.
3. **Eficiència Computacional:** L'ús de paral·lelisme explícit (OpenMP) ha permès un *speedup* de 1.47x respecte a la referència. Tot i que significativa, aquesta millora revela que la CPU Cortex-A53 té els seus límits quan s'enfronta a milions d'operacions de punt flotant.

La comparativa amb l'estàndard industrial (CCSDS 122.0) posa de manifest la realitat tecnològica actual. Els algorismes fixos basats en Wavelets són encara superiors en velocitat pura (12s vs 304s) i simplicitat.

No obstant això, SORTENY ofereix un canvi de paradigma: la capacitat d'adaptar-se. Mentre que el CCSDS és rígid, una xarxa neuronal pot ser reentrenada a terra per a un nou sensor hiperespectral i actualitzada en vol (pujan només els pesos, uns pocs KB), millorant el seu rendiment específic sense canviar el codi de vol. Aquest treball habilita, per primera vegada, aquesta flexibilitat en plataformes de classe CubeSat.

6 TREBALL FUTUR

Aquest projecte obre diverses línies de continuïtat per apropar la solució a un escenari de vol real:

- **Codificador Entròpic:** La tasca pendent més immediata és la integració del codificador aritmètic (Montsec) al mateix executable C, per mesurar ràtios de compressió reals (bits per píxel) i no només estimats.
- **Acceleració Hardware:** L'ús d'instruccions vectorials NEON s'ha delegat al compilador (GCC). Una optimització manual amb intrínsecs NEON o l'ús de biblioteques com *Arm Compute Library* podria reduir significativament el temps d'execució.
- **Validació en Entorn Operatiu (Òrbita):** Es podria elevar el nivell de maduresa tecnològica (TRL). Fent la proposta de portar aquest codi a un model d'enginyeria de CubeSat real per validar-ne el comportament sota condicions de buit tèrmic i avaluar la robustesa del programari davant la radiació (SEUs) en òrbita terrestre baixa (LEO).

DECLARACIÓ D'ÚS D'IA GENERATIVA

En compliment de la normativa d'avaluació de l'assignatura, es declara l'ús d'eines d'intel·ligència artificial generativa en l'elaboració d'aquest Treball de Fi de Grau. Aquest ús s'ha limitat estrictament a tasques de suport instrumental i millora formal. Les eines concretes són les següents:

- **Claude 4.5 Opus (Anthropic) i Gemini 3 Pro (Google):** Utilitzats com a assistents de redacció i motors de cerca semàntica per a documentació tècnica.
- **GitHub Copilot:** Utilitzat dins l'entorn de desenvolupament (IDE) exclusivament per a l'autocompletat predictiu de línies de codi i patrons repetitius.

Aquestes s'han utilitzat per revisar i polir el text generat manualment, amb l'objectiu d'assegurar la correcció lingüística i l'adequació al registre acadèmic. El detall de l'ús és el següent:

- **Ús específic:** Reformulació de paràgrafs per millorar la claredat expositiva, correcció gramatical en català i suport en la traducció de termes tècnics específics a l'anglès per a l'*Abstract*.
- **Grau:** Per suport a la revisió final. El contingut i l'argumentació han estat redactats de forma pròpia.

També en el desenvolupament del programari però s'ha restringit a la consulta sintàctica i la depuració, actuant com una documentació interactiva. Es pot destacar el següent ús:

- **Ús específic:** Consulta sobre la sintaxi de directives de preprocessador complexes en C i *flags* d'optimització de GCC per a l'arquitectura Cortex-A53. Explicació d'errors de compilació obscurs generats per la integració de codi C++ antic (CCSDS).
- **Grau:** Només d'assistència auxiliar. La lògica i metodologia principal, l'algorisme de gestió de memòria (“Ping-Pong”) i l'arquitectura del motor d'inferència han estat dissenyats i implementats manualment.

Aquest ús d'eines d'IA generativa s'ha realitzat amb plena consciència de les implicacions ètiques i acadèmiques, assegurant la transparència i la integritat en tot moment. S'assumeix la responsabilitat total del contingut i la qualitat del treball.

REFERÈNCIES

- [1] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. M. Nasrabadi i J. Chanussot, “Hyperspectral Remote Sensing Data Analysis and Future Challenges”, *IEEE Geoscience and Remote Sensing Magazine*, vol. 1, no. 2, pp. 6–36, 2013. doi:10.1109/MGRS.2013.2244672
- [2] K. Guanter, H. Kaufmann, C. Dobber, et al., “The EnMAP Spaceborne Imaging Spectroscopy Mission for Earth Observation”, *Remote Sensing*, vol. 7, no. 7, pp. 8830–8857, 2015. doi:10.3390/rs70708830

- [3] NASA, “CubeSat 101: Basic Concepts and Processes for First-Time CubeSat Developers”, NASA, 2017. https://www.nasa.gov/wp-content/uploads/2017/03/nasa_csl_i_cubesat_101_508.pdf
- [4] A. Selva i D. Krejci, “A survey and assessment of the capabilities of CubeSats for Earth observation”, *Acta Astronautica*, vol. 74, pp. 50–68, 2012. doi:10.1016/j.actaastro.2011.12.014
- [5] Consultative Committee for Space Data Systems (CCSDS). “Lossless Multispectral and Hyperspectral Image Compression,” CCSDS 123.0-B-2, Blue Book, Washington, D.C., 2019.
- [6] X. X. Zhu, D. Tuia, L. Mou, G.-S. Xia, L. Zhang, F. Xu i J. A. Benediktsson, “Deep Learning in Remote Sensing: A Comprehensive Review and List of Resources”, *IEEE Geoscience and Remote Sensing Magazine*, vol. 5, no. 4, pp. 8–36, 2017. doi:10.1109/MGRS.2017.2762307
- [7] S. Mijares, J. Bartrina-Rapesta, M. Hernández-Cabronero and J. Serra-Sagristà, “Learned Spectral and Spatial Transforms for Multispectral Remote Sensing Data Compression,” in *IEEE Geoscience and Remote Sensing Letters*, vol. 22, pp. 1-5, 2025, Art no. 5001005, doi: 10.1109/LGRS.2025.3554269.
- [8] J. González-de-Regàs, S. Mijares, J. Bartrina-Rapesta and J. Serra-Sagristà, “Robustness of Lossy Multispectral Compression to Simulated Instrumental Noise: A Comparative Study,” in *IEEE Geoscience and Remote Sensing Letters*, vol. 22, pp. 1-5, 2025, Art no. 5003305, doi: 10.1109/LGRS.2025.3623314.
- [9] T. A. Johansen et al., “An Efficient Real-Time FPGA Implementation of the CCSDS-123 Compression Standard for Hyperspectral Images,” in *2018 26th EU-SIPCO*, 2018.
- [10] Consultative Committee for Space Data Systems (CCSDS). “Lossless Data Compression,” CCSDS 121.0-B-3, Blue Book, Washington, D.C., 2020.
- [11] L. Santos, A. Gomez, and R. Sarmiento, “Implementation of CCSDS Standards for Lossless Multispectral and Hyperspectral Satellite Image Compression,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 56, no. 3, pp. 2120-2133, 2020.
- [12] D. Báscones, C. González, and D. Mozos, “Parallel Implementation of the CCSDS 1.2.3 Standard for Hyperspectral Lossless Compression,” *Remote Sensing*, vol. 9, no. 10, 2017.
- [13] Consultative Committee for Space Data Systems (CCSDS). “Image Data Compression,” CCSDS 122.0-B-1, Blue Book, Washington, D.C., 2005. [Online]. Available: <https://web.archive.org/web/20110718053241/http://public.ccsds.org/publications/archive/122x0b1c2.pdf>.
- [14] Consultative Committee for Space Data Systems (CCSDS). “Image Data Compression,” CCSDS 122.0-B-2, Blue Book, Washington, D.C., 2017.
- [15] P. Zicter et al., “A 13.3 Gbps 9/7M Discrete Wavelet Transform for CCSDS 122.0-B-1 Image Data Compression on a Space-Grade SRAM FPGA,” *Electronics*, vol. 9, no. 8, 2020.
- [16] J. I. Agudo-G. et al., “Discrete wavelet transform fully adaptive prediction error coder: Image data compression based on CCSDS 122.0,” *IEEE Latin America Trans.*, vol. 13, no. 10, 2015.
- [17] Raspberry Pi Ltd., “Raspberry Pi 3 Model B+ Product Brief,” 2018. [Online]. Available: <https://datasheets.raspberrypi.com/rpi3/raspberry-pi-3-b-plus-product-brief.pdf>.
- [18] Xilinx Inc., “Zynq UltraScale+ MPSoC Data Sheet: Overview (DS891),” v1.9, 2023. [Online]. Available: <https://docs.amd.com/v/u/en-US/ds891-zynq-ultrascale-plus-overview>.
- [19] B. Schwaller, B. Ramesh and A. D. George, “Investigating TI KeyStone II and quad-core ARM Cortex-A53 architectures for on-board space processing,” 2017 IEEE High Performance Extreme Computing Conference (HPEC), Waltham, MA, USA, 2017, pp. 1-7, doi: 10.1109/HPEC.2017.8091094.
- [20] S. M. Guertin, “Raspberry Pis for Space Guidelines,” NASA Jet Propulsion Laboratory (JPL), 2021. [Online]. Available: <https://nepp.nasa.gov/docs/papers/2021-Guertin-Raspberry-Pi-Guideline-CL-21-564.pdf>.

APÈNDIX

A.1 Figures

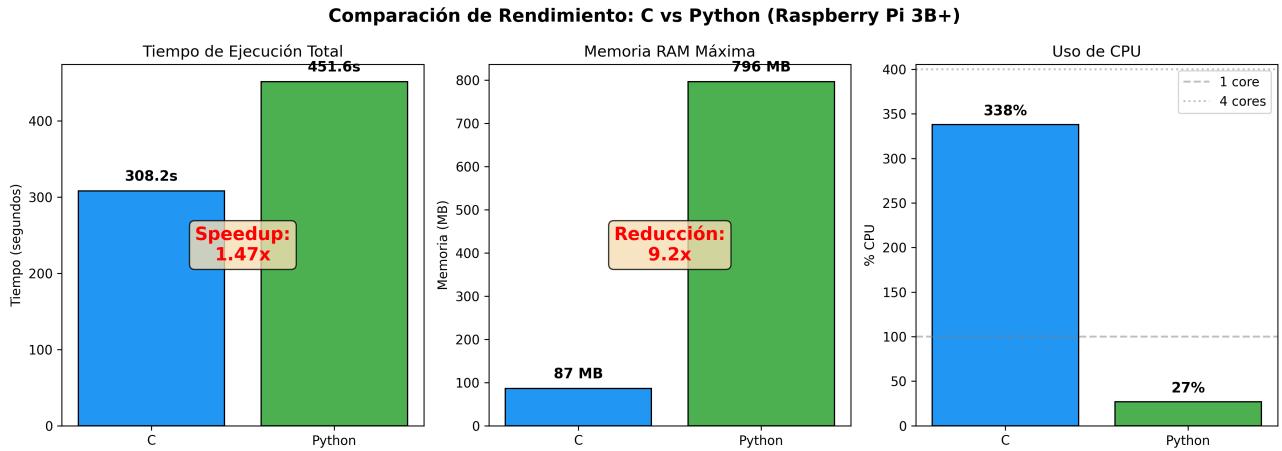


Fig. 5: Comparativa de rendiment en Raspberry Pi 3B+. (Esquerra) Temps d'execució total. (Centre) Consum màxim de RAM. (Dreta) Ús percentual de CPU (multifil).

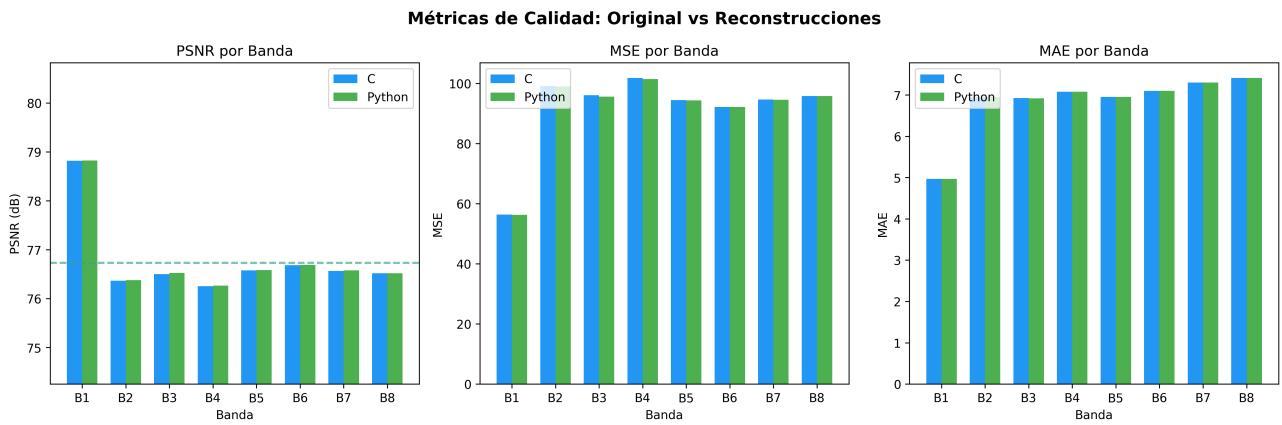


Fig. 6: Mètriques de qualitat (PSNR) per banda espectral. S'observa una variació natural de la qualitat dependent del contingut espectral (millor en Blau/Verd, pitjor en NIR), però idèntica entre implementacions.

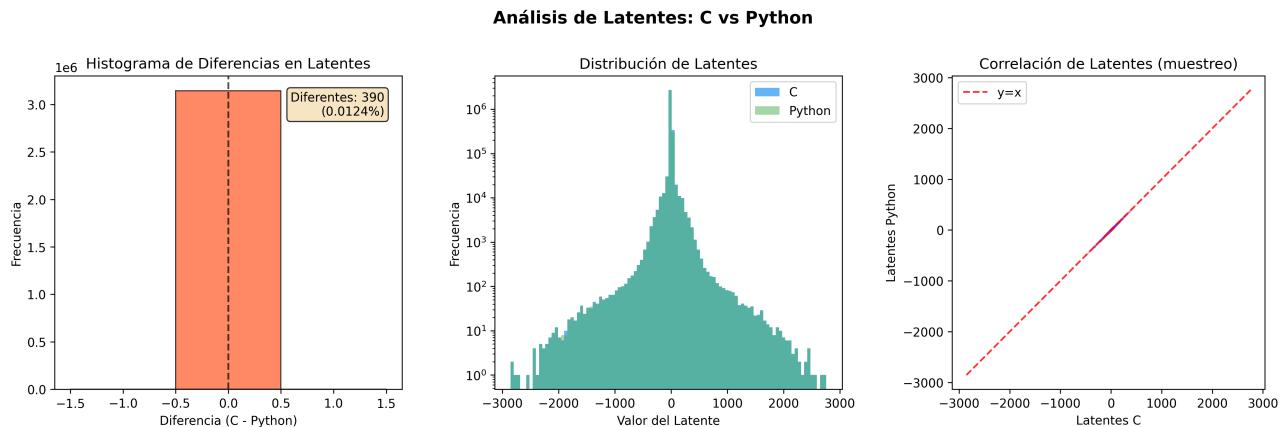


Fig. 7: Anàlisi de paritat. L'histograma de diferències (esquerra) mostra que la immensa majoria de valors són idèntics (diferència 0), amb variacions residuals de ± 1 degudes a l'arrodoniment.

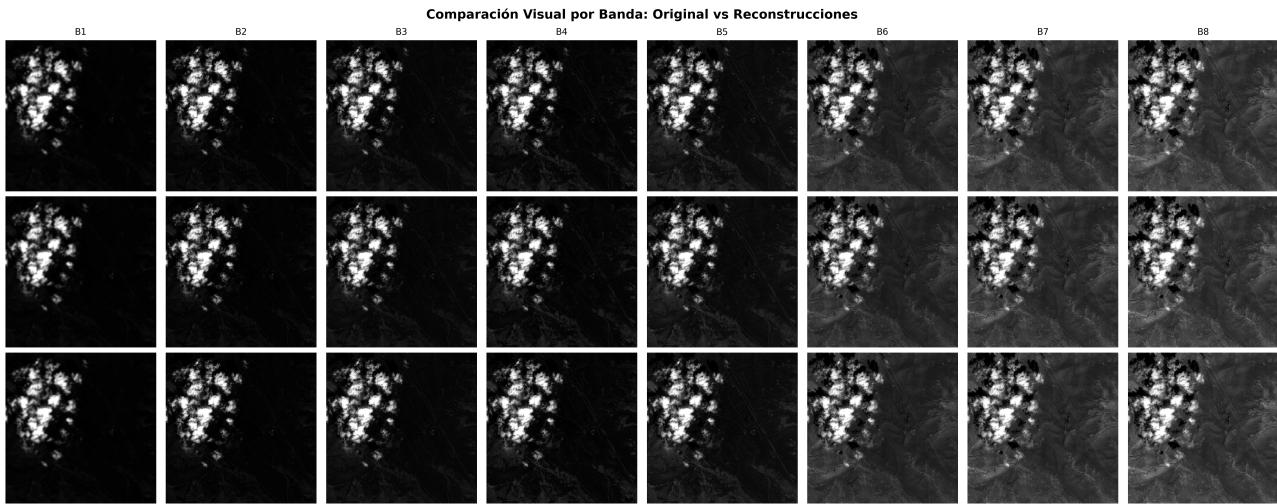


Fig. 8: Comparativa visual de les bandes reconstruïdes. (Dalt) Imatge Original. (Mig) Reconstrucció SORTENY C. (Baix) Reconstrucció SORTENY Python. Visualment indistingibles.

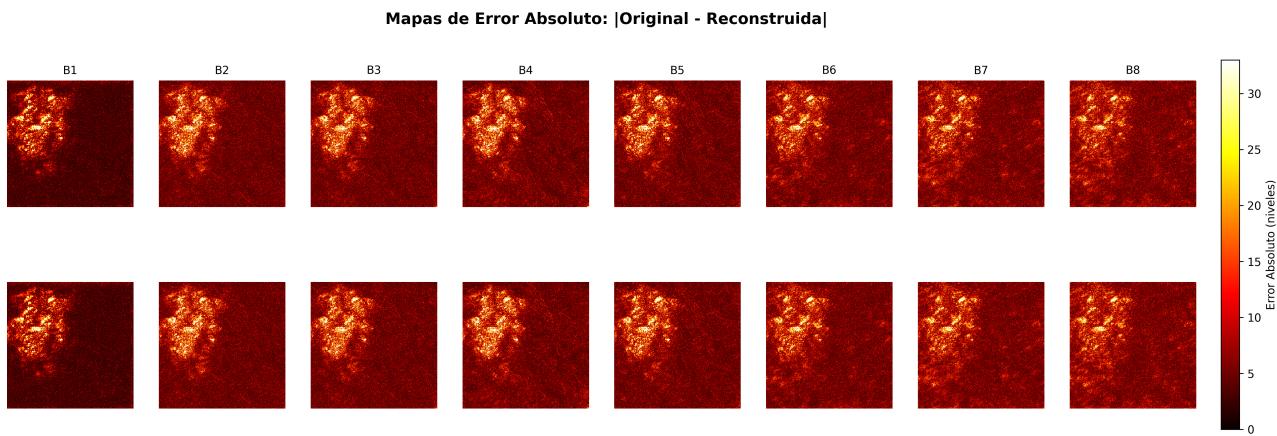


Fig. 9: Mapes d'error absolut $|Original - Reconstruida|$ per banda. Fila superior: error des de C. Fila inferior: error des de Python. L'escala de color (hot) mostra major error en colors més clars. L'error és uniforme i baix, sense patrons anòmals.

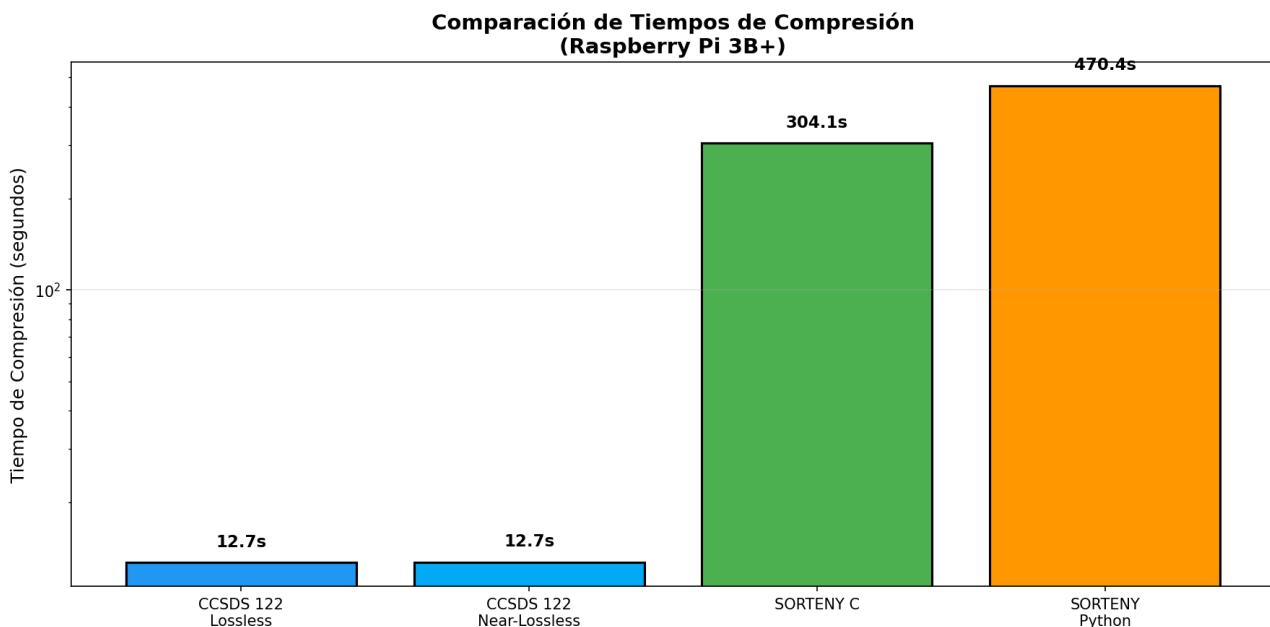


Fig. 10: Temps de compressió (Escala Logarítmica). CCSDS és un ordre de magnitud més ràpid.