
This is the **published version** of the bachelor thesis:

Blesa Rodriguez, Arnau. *Desenvolupament en CUDA d'una tècnica de compressió basada en IA*. Treball de Final de Grau (Universitat Autònoma de Barcelona), 2026 (Enginyeria Informàtica)

This version is available at <https://ddd.uab.cat/record/334431>

under the terms of the  license.

Desenvolupament en CUDA d'una tècnica de compressió basada en IA

Arnau Blesa Rodríguez

11 de juny de 2026

Resum– La *Reversible Regression Network* (R2-Net) és un model de predicció intra-bloc basat en DPCM que, tot i eliminar la transmissió de paràmetres, imposa una dependència causal estricta que provoca *GPU Starvation*. Aquest treball proposa dues estratègies per superar aquest coll d'ampolla: el mètode *Wavefront*, que explota el paral·lelisme diagonal preservant les dependències causals, i el mètode *Parallel*, que les elimina per vectoritzar la imatge sencera en una sola operació. Implementades sobre CPU i GPU mitjançant CUDA i CuPy, s'assoleix un *speedup* màxim de 583.8×. S'analitza també l'impacte de la precisió numèrica, demostrant que `float32` duplica el rendiment amb una pèrdua de qualitat imperceptible (PSNR de 75.87 dB), i que el mètode *Parallel* penalitza la compressibilitat teòrica en trencar la correlació espacial entre blocs.

Paraules clau– Compressió d'imatges, R2-Net, DPCM, CUDA, Wavefront, Paral·lelisme, Speedup, Entropia.

Abstract– The *Reversible Regression Network* (R2-Net) is an intra-block prediction model based on DPCM that eliminates parameter transmission but imposes a strict causal dependency between blocks, causing *GPU Starvation*. This work proposes two strategies to overcome this bottleneck: the *Wavefront* method, which exploits diagonal parallelism while preserving causal dependencies, and the *Parallel* method, which removes them to vectorize the entire image in a single operation. Both are implemented on CPU and GPU via CUDA and CuPy, achieving a maximum speedup of 583.8×. Numerical precision is also analyzed, showing that `float32` doubles throughput with negligible quality loss (PSNR of 75.87 dB), while the *Parallel* method penalizes theoretical compressibility by breaking spatial correlation between blocks.

Keywords– Image compression, R2-Net, DPCM, CUDA, Wavefront, Parallelism, Speedup, Entropy.



1 INTRODUCCIÓ I CONTEXT

AQUEST treball aborda la problemàtica del cost computacional associat a les xarxes neuronals aplicades a la compressió d'imatges. En concret, s'analitza la *Reversible Regression Network* (R2-Net) [1], un model de predicció intra-bloc que millora substancialment el rendiment de compressió al eliminar la necessitat de transmetre els paràmetres del model, però que imposa una dependència espacial estricta a causa de la seva naturalesa basada en *Differential Pulse Code Modulation* (DPCM).

Aquesta restricció força un recorregut seqüencial que limita greument el rendiment en arquitectures de computació paral·lela, provocant la inanició de nuclis o *GPU Starvation*.

Per dur a terme aquesta anàlisi, el projecte es vertebrava en quatre objectius: **i) Estudi teòric i validació de la referència** mitjançant el desenvolupament de la implementació seqüencial original proposada a la R2-Net en CPU (*baseline*); **ii) Disseny de models de recorregut paral·lel**; **iii) Desenvolupament i programació multinucli** en entorns multi-fil i posterior migració a *kernels* CUDA; i **iv) Avaluació experimental de rendiment** temporal i de l'impacte de la precisió de coma flotant (*float64*) en la reversibilitat.

El document s'estructura de la manera següent: la Secció 2 presenta el marc teòric de la R2-Net; la Secció 3 descriu la metodologia; la Secció 4 detalla les estratègies de paral·lelització proposades (*Wavefront* i *Parallel*) sobre CPU i GPU; la Secció 5 analitza els resultats experimentals; i la Secció 6 recull les conclusions i les línies de treball futur.

- E-mail de contacte: ablesarodriguez@gmail.com
- Menció: Tecnologies de la Informació
- Treball tutoritzat per: Joan Bartrina (Departament d'Enginyeria de la Informació i les Comunicacions)
- Curs 2025/26

2 MARC TEÒRIC: LA XARXA R2-NET

La *Reversible Regression Network* (R2-Net) [1] és una xarxa neuronal superficial completament connectada d'una sola capa (*1-layer FC-NN*) (veure la Fig. 1) dissenyada per a la predicció intra-bloc de dades d'imatge. A diferència dels models clàssics ponderats per un entrenament *offline*, la R2-Net s'optimitza completament *online* mitjançant un sobreajustament (*overfitting*) de només dues iteracions de retropropagació per cada bloc individual. Això elimina totalment la necessitat de transmetre els paràmetres del model en el *bit-stream*, ja que el descodificador és capaç de reconstruir analíticament les dades originals a partir del residu rebut.

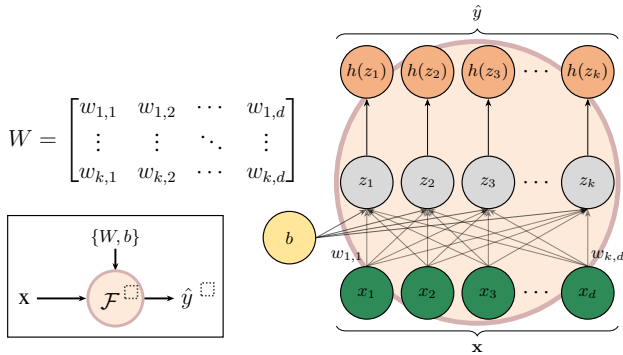


Fig. 1: Xarxa neuronal completament connectada (FC-NN) d'una capa amb el vector de característiques d'entrada x i el vector de sortida $\hat{y} = h(z)$, on $h(\cdot)$ és una funció d'activació no lineal. La matriu $W = [w_{m,n}] \in \mathbb{R}^{k \times d}$ i el vector de biaix $b = [b_n] \in \mathbb{R}^k$ són els paràmetres de la xarxa.

2.1 Funcionament i Formulació Analítica

El model rep com a entrada un vector $x \in \mathbb{R}^d$ amb els píxels de context circumdants (detallat en la següent Subsecció 2.2, *podeu veure la Fig. 3 per entrar en context*), i prediu un bloc $g \in \mathbb{R}^k$ organitzat com un vector columna. El procés d'optimització i el flux reversible s'estructuren en les dues fases seqüencials que es detallen pas a pas a la mateixa Fig. 2:

1. **Iteració $t = 0$:** Es calcula la predicció inicial $\hat{y}^0 = \sigma(W^0 x + b^0)$ i el seu residu intermedi $e^0 = g - \hat{y}^0$. Els pesos s'actualitzen invertint el signe del gradient per deslligar el residu final de la veritat de referència (g):

$$W^1 = W^0 + \alpha e^0 x^T, \quad b^1 = b^0 + \alpha e^0 \quad (1)$$

2. **Iteració $t = 1$:** Es genera la predicció refinada $\hat{y}^1 = \sigma(W^1 x + b^1) + e^0$. L'error residual final transmès al flux de bits és $e^1 = \hat{y}^1 - g$, el qual algebraicament es simplifica com:

$$e^1 = \sigma(W^1 x + b^1) - \hat{y}^0 \quad (2)$$

En recepció, atès que W^0, b^0 i x són coneguts pel receptor, el descodificador calcula $\hat{y}^0$ i aïlla analíticament l'error latent e^0 aplicant la funció **Logit** (σ^{-1}) com a inversa:

$$e^0 = \frac{\sigma^{-1}(e^1 + \hat{y}^0) - (W^0 x + b^0)}{\alpha(x^T x + 1)} \quad (3)$$

Finalment, el bloc original es reconstrueix exactament com a $\hat{g} = e^0 + \hat{y}^0$.

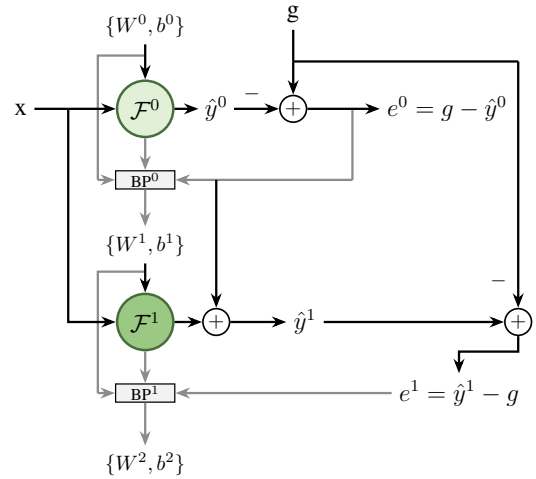


Fig. 2: Càlculs detallats de la xarxa R2-Net per a dues iteracions de retropropagació (BP^0 i BP^1), on es mostra la dependència temporal i el flux de dades reversible.

2.2 Dependència Espacial i Coll d'Ampolla

L'estructura del vector d'entrada x depèn de forma estricta dels píxels veïns immediatament anteriors ja reconstruïts. Aquesta causalitat espacial, heretada de la naturalesa DPCM, imposa un processament obligatori en ordre *Raster Scan* (Fig. 3).

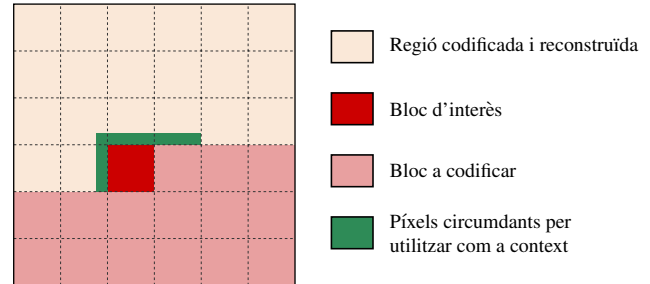


Fig. 3: Predicció intra-bloc de la R2Net: Els blocs es prediuen seqüencialment seguint un ordre específic. Noteu que, per als blocs situats a les vores superior ($i = 0$) i esquerra ($j = 0$), part del context espacial excedeix els límits de la imatge.

Per gestionar el cas dels blocs situats a les vores, on part del context espacial recau fora del domini vàlid de la imatge, s'assigna a aquestes posicions un valor fix de 0.5. Aquest valor correspon al punt mitjà de l'interval de normalització $[0, 1]$.

A més d'aquesta dependència espacial, la formació del vector d'entrada x inclou un procés de submostreig (*down-sampling*) mitjançant la mitjana dels píxels de referència. Concretament, en lloc d'utilitzar directament tots els píxels adjacents, aquests s'agrupen en conjunts no superposats per calcular-ne la mitjana i establir la mida del vector de característiques d'entrada a un valor fix (per exemple, `FIXED_INPUT_SIZE = 16`). Aquesta estratègia compleix dues funcions fonamentals: en primer lloc, redueix signi-

ficativament el nombre de paràmetres que la xarxa ha d'aprendre; en segon lloc, ajuda a eliminar el soroll present als píxels de referència, la qual cosa s'ha demostrat que millora la precisió de la predicció.

2.3 El Bucle del Codificador: Feedback Loop

Atès que l'algorisme emprà una arquitectura predictiva utilitzant un DPCM, el vector de context $\mathbf{x}$ ha de ser idèntic en l'etapa de codificació i descodificació. Si el codificador extragués el context de la imatge original ($\mathbf{x}_{orig}$) i el descodificador remot de la reconstruïda ($\mathbf{x}_{rec}$), el residu transmès $\mathbf{e} = \mathbf{g} - \mathcal{M}(\mathbf{x}_{orig})$ generaria una divergència acumulada (*drift*) en la imatge final:

$$\hat{\mathbf{g}}_{dec} - \mathbf{g} = \mathcal{M}(\mathbf{x}_{rec}) - \mathcal{M}(\mathbf{x}_{orig})$$

A causa de les limitacions de precisió numèrica (*float64*), $\mathbf{x}_{orig} \neq \mathbf{x}_{rec}$, fet que amplificaria l'error recursivament per tota la imatge. Per anul·lar aquest terme, s'imposa un descodificador local integrat: el codificador s'obliga a extreure el context exclusivament del búfer reconstruït $\mathbf{I}_{rec}$. D'aquesta manera es garanteix analíticament que $\mathcal{M}(\mathbf{x}_{rec}) - \mathcal{M}(\mathbf{x}_{rec}) = 0$, blindant la reversibilitat del sistema.

3 METODOLOGIA D'IMPLEMENTACIÓ

3.1 Vectorització i Adaptació Matricial del Model

L'arquitectura original de la R2-Net està dissenyada per processar el context com un vector unidimensional $\mathbf{x} \in \mathbb{R}^K$, operant de forma estrictament seqüencial bloc a bloc. No obstant això, per aprofitar el paradigma d'execució paral·lela massiva (SIMT) de les GPU mitjançant CUDA, ha estat necessari refactoritzar completament el nucli matemàtic de la xarxa abans d'implementar les estratègies de recorregut.

La metodologia aplicada ha consistit en la vectorització de tota la lògica d'inferència i retropropagació original. En lloc d'avaluar les equacions d'actualització de la Secció 2.1 per a una instància aïllada, el model de programació s'ha adaptat per rebre tensors d'entrada que agrupen N blocs independents, conformant un lot de dades (*batch processing*).

Aquesta reestructuració substitueix els càlculs iteratius per operacions de producte matricial concurrent al llarg d'una nova dimensió de profunditat. D'aquesta manera, l'avaluació de la funció d'activació sigmoide, el càlcul de residus i l'actualització de la matriu de pesos W i del biaix b s'executen simultàniament per a múltiples blocs en una única crida a la memòria del dispositiu.

3.2 Farciment de Vores (*Edge Padding*)

L'algorisme de compressió opera sobre la imatge iterant en blocs no solapats de mida fixa $B \times B$ (per exemple, $B = 16$, on B és la quantitat de píxels). Si les dimensions espacials originals de la imatge, denotades per la seva alçada H i amplada W , no són múltiples exactes de la mida del bloc B , la divisió en la quadrícula deixarà dades incompletes als límits inferior i dret.

Per solucionar-ho, s'utilitza la tècnica de *edge padding* [5], la qual consisteix a copiar i replicar repetidament els píxels de la vora de la imatge original I . Cal destacar que aquest procés d'adequació dimensional s'aplica de manera universal com a pas previ per a totes les implementacions de l'algorisme (Seqüencial, *Wavefront* i Paral·lel).

Aquesta decisió estableix una base estructural homogènia per a tots els mètodes i està motivada per l'arquitectura del maquinari, que processa de manera molt més eficient els blocs de mida fixa evitant divergències en l'execució. A conseqüència d'aquest farciment, és estrictament necessari transmetre les dimensions originals de la imatge dins del *bit-stream*, permetent així al descodificador aplicar el retall corresponent per recuperar la resolució autèntica un cop finalitzada la reconstrucció.

4 ESTRATÈGIES DE PARAL·LELITZACIÓ

4.1 Algorisme Wavefront

En aquesta secció es detalla l'estratègia de processament paral·lel implementada per a la xarxa R2Net, coneguda com a mètode *Wavefront* [7]. L'algorisme *Wavefront* resol la limitació de la R2Net mitjançant una reordenació geomètrica de l'execució: en lloc de processar la imatge fila per fila, l'avaluació avança a través de les anti-diagonals de la quadrícula de blocs, tal com s'aprecia a la Fig. 4. Atès que els blocs situats en una mateixa diagonal no tenen dependències de dades entre si, aquests poden ser codificats i reconstruïts simultàniament, maximitzant així el paral·lelisme i l'ús dels recursos computacionals.

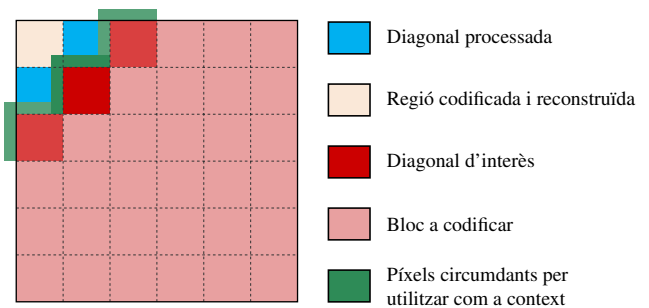


Fig. 4: Predicció intra-bloc del mètode *Wavefront*: El processament es realitza en diagonals. Per a cada bloc d'interès, el context espacial s'extreu en forma de «L» invertida a partir de les diagonals anteriors que ja han estat codificades i reconstruïdes. Noteu que, per als blocs situats a la vora superior ($i = 0$) o esquerra ($j = 0$), part dels píxels del context espacial queden fora del domini vàlid de la imatge; procediment detallat en aquesta secció.

4.1.1 Inicialització i Gestió de les Condicions de Contorn

Matemàticament, si definim la imatge a processar com una matriu $\mathbf{I} \in \mathbb{R}^{H \times W}$, on H i W són l'alçada i l'amplada respectivament, cada bloc de mida $b \times b$ situat a la coordenada d'origen (i, j) requereix accedir als píxels reconstruïts de les posicions adjacents anteriors. Això planteja un problema de condicions de contorn per als blocs situats a la primera fila ($i = 0$) o a la primera columna ($j = 0$), ja que el

seu context espacial cau fora del domini vàlid de la imatge.

Per resoldre aquesta singularitat de manera eficient i vectoritzada, es defineix una matriu de reconstrucció estesa o *padding buffer*, denotada com $\mathbf{R}_{pad} \in \mathbb{R}^{(H+1) \times (W+1)}$. Aquesta matriu incorpora una vora addicional d'1 píxel a la part superior i esquerra. Abans d'iniciar la compressió, la matriu s'inicialitza completament amb un valor constant:

$$\mathbf{R}_{pad}(i, j) = 0.5, \quad \forall i \in [0, H], \forall j \in [0, W]$$

Aquest valor neutral de 0.5 s'escull perquè les dades d'entrada a la xarxa estan normalitzades en el rang $[0, 1]$. Assumir un valor central minimitza l'error esperat i evita introduir gradients artificials severes en les vores de la imatge quan no es disposa d'informació prèvia.

4.1.2 Propagació del Front d'Ona i Identificació de Blocs

L'estratègia *Wavefront* processa la imatge a través de les seves anti-diagonals, on cadascuna s'identifica per un índex k . El gran avantatge d'aquest mètode és que els blocs d'una mateixa diagonal compleixen la condició geomètrica $i + j = k$. Com que les seves coordenades (delimitades per l'expressió $i \in [\max(0, k - N_{cols} + 1), \min(N_{rows} - 1, k)]$) demostren que no depenen els uns dels altres per formar el seu context, es poden processar simultàniament en paral·lel.

Extracció Vectoritzada del *Ground Truth* En lloc de recórrer la imatge bloc a bloc de forma seqüencial, l'algorisme calcula de cop les coordenades matricials inicials de tota la diagonal mitjançant els vectors $\mathbf{r}_s$ i $\mathbf{c}_s$. En escalar aquests valors per la mida del bloc b , s'obtenen les coordenades espacials absolutes $\mathbf{p}_y$ i $\mathbf{p}_x$ tal com es veu a la Fig. 5.

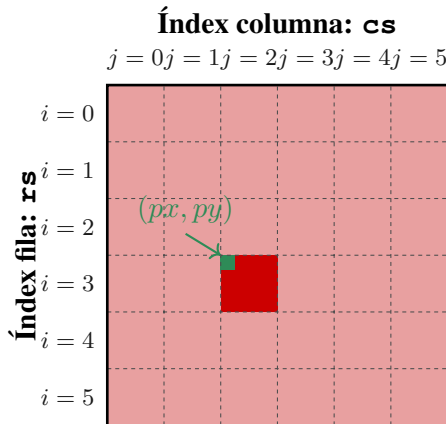


Fig. 5: Diferenciació estructural entre índexs de blocs i coordenades físiques de píxels: Els vectors $\mathbf{r}_s$ i $\mathbf{c}_s$ actuen com a localitzadors ordinals discretitzats (i, j) dins la matriu de blocs. En canvi, $\mathbf{p}_y$ i $\mathbf{p}_x$ defineixen la posició mètrica real del primer píxel.

A partir d'aquests punts d'origen, s'aplica un desplaçament local bidimensional representat per $\Delta = [0, 1, \dots, b - 1]^T$. Aquesta operació genera les graelles de coordenades ($\mathbf{R}_{grid}$ i $\mathbf{C}_{grid}$) necessàries per capturar simultàniament tots els píxels interns de les àrees

d'interès. Això genera un paquet de dades unificat (el tensor *Ground Truth*, $\mathbf{G}^{(k)} \in \mathbb{R}^{N \times b^2 \times 1}$) que queda llest per ser processat per la xarxa neuronal en una sola operació. *Explicació detallada a l'apèndix A.1.*

4.1.3 Extracció del Context Espacial

Per poder processar qualsevol bloc, la xarxa necessita la informació dels píxels veïns que ja han estat reconstruïts prèviament. Aquests píxels formen una topologia causal en forma de «L» invertida. Per extreure aquest context de la manera més eficient i simple possible, s'utilitza la matriu de reconstrucció estesa $\mathbf{R}_{pad}$, la qual disposa d'un marge addicional (*padding*) a la part superior i esquerra. Aquest marge actua de forma natural com un desplaçament automàtic: si s'indexa $\mathbf{R}_{pad}$ utilitzant les coordenades originals $(\mathbf{p}_y, \mathbf{p}_x)$, el sistema no llegeix l'interior del bloc, sinó que recull directament els píxels veïns que el rodegen. Aprofitant aquesta propietat, s'extreuen fàcilment tres segments adjacents (esquerre, cantonada i superior), que es concatenen espacialment per formar un únic tensor de context global sense necessitat de programar condicions de contorn complexes:

$$\mathbf{X}_{all} = [\mathbf{X}_{left} \quad \mathbf{X}_{tl} \quad \mathbf{X}_{top}]$$

Atès que la longitud d'aquest context extret (de mida $L = 2b + 1$ píxels) sol ser superior a la capacitat d'entrada requerida per la xarxa (K), s'aplica una reducció dimensional (*downsampling*). Això s'aconsegueix de forma vectoritzada multiplicant el tensor $\mathbf{X}_{all}$ per una matriu de projecció precalculada $\mathbf{M}$ (veure A.2), la qual agrupa i fa la mitjana dels píxels automàticament per generar el vector d'entrada definitiu:

$$\mathbf{X}_{batch} = \mathbf{X}_{all} \cdot \mathbf{M}$$

4.1.4 Predicció, Reconstrucció i Actualització de Tensors

Amb el tensor de dades originals $\mathbf{G}^{(k)}$ i el de context $\mathbf{X}_{all}$ preparats, la xarxa neuronal genera la predicció $\hat{\mathbf{Y}}^{(k)}$ per a tota la diagonal de manera concurrent. A continuació, es calcula l'error residual (la diferència entre l'original i la predicció) algebraicament com $\mathbf{E}^{(k)} = \mathbf{G}^{(k)} - \hat{\mathbf{Y}}^{(k)}$, i s'inverteix el procés per obtenir el tensor de dades reconstruïdes $\hat{\mathbf{G}}^{(k)}$.

Finalment, per preparar la següent iteració, aquests tensors s'actualitzen a la memòria global de forma diferenciada: el mapa d'errors s'escriu sobre les coordenades exactes de la imatge original ($\mathbf{R}_{grid}, \mathbf{C}_{grid}$), mentre que els nou blocs reconstruïts es guarden a la matriu estesa $\mathbf{R}_{pad}$ aplicant un desplaçament d'una unitat ($\mathbf{R}_{grid} + 1, \mathbf{C}_{grid} + 1$). Així, quan l'algorisme avanci cap a la següent anti-diagonal, el context estarà automàticament a lloc per continuar el procés sense interrupcions.

4.1.5 Simetria del Procés de Descompressió

Gràcies al bucle de reconstrucció local del codificador (Secció 2.3), el procés de descompressió és computacionalment simètric. A partir de les dimensions originals i el mapa de residus rebut, el receptor inicialitza el bufer $\mathbf{R}_{pad}$ i replica el mateix recorregut en *Wavefront*. Per a cada diagonal k ,

s'extreu el context, la xarxa prediu $\hat{\mathbf{Y}}^{(k)}$ i es recuperen les dades exactes sumant-hi l'error:

$$\hat{\mathbf{G}}^{(k)} = \hat{\mathbf{Y}}^{(k)} + \mathbf{E}^{(k)}$$

Aquesta equivalència estructural permet reaprofitar de manera transparent tota la vectorització i l'acceleració GPU que es detallen en la secció següent. El procés conclou amb una simple operació de retall (*cropping*) per eliminar el *padding* i restaurar la resolució nativa de la imatge.

4.1.6 Transició a CUDA mitjançant CuPy

L'acceleració de les equacions matricials s'ha implementat sobre l'arquitectura CUDA mitjançant la llibreria CuPy [2]. Aquesta decisió preserva l'API de referència de NumPy, garantint una translació algebraica estricta del model teòric, alhora que compila les instruccions *Just-In-Time* (JIT) per explotar el paradigma SIMT (*Single Instruction, Multiple Threads*) de la GPU [11].

L'adaptació computacional es fonamenta en tres optimitzacions de baix nivell:

Primer, l'**allotjament estàtic de memòria**: s'elimina la sobrecàrrega d'assignació dinàmica pre-assignant de forma contigua a la memòria de vídeo (`cp.empty`) els tensors de la xarxa, limitant les transferències *Host-to-Device* exclusivament a l'entrada i sortida de dades.

Segon, la **contracció tensorial nativa**: els bucles iteratius d'inferència se substitueixen per operacions de producte matricial per lots (`cp.matmul`) [6], transformant el càlcul seqüencial en avaluacions algebraiques sobre tensors tridimensionals que aprofiten la jerarquia de memòria cau del maquinari.

Finalment, la **fusió de kernels via JIT**: per mitigar la latència dels accessos redundants a la memòria global, el compilador empaqueta les avaluacions elementals (com l'activació sigmoide) en un únic micro-programa (`@cp.fuse`) [8]. Aquesta reducció d'operacions I/O maximitza la intensitat aritmètica del dispositiu abans de transferir l'`error_map` resultant de tornada a la CPU (`cp.asnumpy`).

4.2 Algorisme Parallel

L'estratègia *Parallel* porta la vectorització al seu límit teòric transformant el problema de resolució espai-temporal d'ordre $O(H/b + W/b)$ del mètode *Wavefront* en una única operació tensorial global d'ordre $O(1)$. Per habilitar aquesta simultaneïtat total, es trenca axiomàticament la dependència causal inter-bloc. Tal com s'il·lustra a la Fig. 6, el subespai de context $\mathbf{x}$ ja no s'extreu dels blocs adjacents reconstruïts (d'acord amb la Secció 4.1.1), sinó que s'obté exclusivament de la frontera interior del propi bloc d'interès.

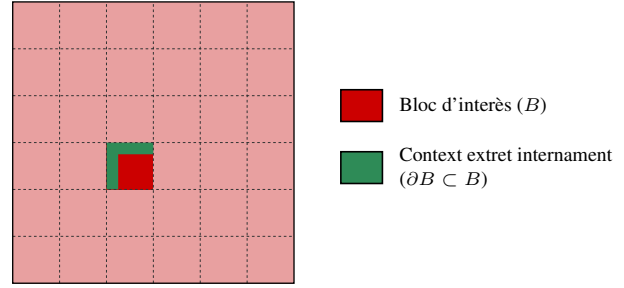


Fig. 6: Predicció intra-bloc del mètode *Parallel*: El processament és totalment independent per a cada bloc. A diferència del *Wavefront*, no hi ha dependència de veïns; el context espacial s'extreu dels píxels de la frontera interior del propi bloc d'interès.

4.2.1 Transformació Isomòrfica i Extracció Massiva

L'enfocament *Parallel* abandona la lògica iterativa basada en diagonals del *Wavefront* per adoptar una estratègia de processament monolítica. Mentre que en el *Wavefront* l'extracció de context és un procés dinàmic que depèn del progrés del front d'ona, en el mètode *Parallel* la imatge sencera es projecta cap a un espai tensorial de blocs independents abans de començar la inferència. Aquest procés es formalitza mitjançant una transformació $\mathcal{T}$ que mapeja la imatge original $\mathbf{I}_{pad} \in \mathbb{R}^{H \times W}$ cap a un tensor de blocs $\mathbf{B} \in \mathbb{R}^{N \times b \times b}$, essent $N = (H/b) \times (W/b)$.

L'operació es realitza en tres fases algebraiques que manipulen la topologia de les dades sense necessitat de moure píxels individualment:

1. **Descomposició Espacial (4D)**: La matriu bidimensional es projecta a un espai de quart ordre $\mathbb{R}^{\frac{H}{b} \times b \times \frac{W}{b} \times b}$. Aquesta operació separa explícitament les magnituds macroscòpiques (índexs de la quadrícula de blocs) de les microscòpiques (coordenades del píxel dins el bloc). L'eix 1 representa l'alçada del bloc (Y) i l'eix 3 l'amplada (X).
2. **Permutació Ortogonal**: S'aplica una transposició $\mathcal{P}(0, 2, 1, 3)$. Aquest és el pas crític: reordena els eixos de memòria de manera que les coordenades de la quadrícula (Grid Y, Grid X) ocupin els dos primers eixos, mentre que les coordenades del píxel (Pixel Y, Pixel X) es reclusen en els dos últims. Aquesta operació modifica els *strides* de memòria, convertint els salts discontinus de les files de la imatge original en blocs contigus.
3. **Aplanament del Lot**: Finalment, es col·lapsen les dimensions de la quadrícula en un únic eix de dimensió N . El resultat és un tensor unificat on cada entrada $n \in N$ és una matriu $b \times b$ perfecta i contigua.

La superioritat d'aquest mètode respecte al *Wavefront* rau en la seva **localitat de dades**. Mentre que el *Wavefront* pateix de salts de memòria no alineats durant l'extracció de les diagonals, el mètode *Parallel* garanteix que el controlador de memòria accedeixi a blocs geomètricament quadrats com si fossin vectors lineals. Això permet que l'inferència de la R2-Net i el càlcul del residu $\mathbf{E} = \mathbf{G} - \hat{\mathbf{Y}}$ es realitzin amb una única operació `matmul` per a tota la imatge,

minimitzant les latències d'accés a memòria global i maximitzant la saturació dels nuclis CUDA.

4.2.2 DPCM i Context Local

L'arquitectura *Parallel* imposa una restricció fonamental: a diferència del mètode *Wavefront*, on el context $\mathbf{x}$ s'extreu dinàmicament dels píxels veïns ja reconstruïts i disponibles a la memòria global ($\mathbf{R}_{pad}$), en el mètode *Parallel* cada bloc és una unitat d'execució totalment aïllada. En no tenir accés a la informació d'altres blocs durant la fase d'inferència, el sistema es veu obligat a generar el seu propi context a partir dels píxels interns de la frontera del bloc (∂B).

Tanmateix, la simple extracció de la vora ∂B no seria suficientment eficient per a la R2-Net. Si la xarxa rebés directament les intensitats dels píxels de vora sense cap tipus de processament, la redundància espacial intrínseca de les dades (*inter-pixel correlation*) dificultaria l'aprenentatge del model. Per tal de mitigar aquesta limitació i garantir que el context contingui informació rellevant i descorrelacionada, s'aplica una transformació DPCM (*Differential Pulse Code Modulation*) sobre la vora $\partial B = \{v_0, v_1, \dots, v_{b-1}\}$. La frontera es codifica mitjançant gradients de primer ordre:

$$\nabla \mathbf{v}_i = \begin{cases} v_0 & \text{si } i = 0 \\ v_i - v_{i-1} & \text{si } i \in [1, b-1] \end{cases} \quad (4)$$

Aquest mecanisme permet que cada bloc sigui capaç de sintetitzar una representació del context robusta sense necessitar dades externes, transformant les intensitats absolutes en variacions locals que la xarxa processa de manera més estable. Després, el receptor, en processar el mapa d'errors $\mathbf{E}$, pot recuperar la vora ∂B mitjançant l'operador invers de suma acumulada. Com que aquest procés es realitza localment per a cada bloc, es manté la coherència amb el paradigma de paral·lelisme total, garantint que $\mathbf{R}_{rec} = \sum \nabla \mathbf{v}_i$ sigui idèntic al senyal original sense requerir comunicació inter-bloc.

4.2.3 Reducció Dimensional i Mapeig

El vector d'entrada $\mathbf{x} \in \mathbb{R}^L$ es construeix a partir de ∂B , resultant en una longitud $L = 2b - 1$ (ja que la cantonada superior-esquerra no es repeteix). S'aplica el mateix producte matricial contra la matriu de projecció $\mathbf{M}$ (definida a l'apèndix A.2) per ajustar la dimensió a la capa d'entrada de la R2-Net:

$$\mathbf{X}_{batch} = \mathbf{X}_{all} \cdot \mathbf{M}$$

La màscara de validesa `valid_groups` garanteix que les projeccions buides s'inicialitzin a 0.5, mantenint la coherència del flux de dades $\mathbf{X}_{batch} \in \mathbb{R}^{N \times K \times 1}$.

4.2.4 Inferència Global i Reensamblatge

Amb el tensor de dades originals $\mathbf{G}_{batch}$ i el context $\mathbf{X}_{batch}$ a la mateixa profunditat N , la xarxa neuronal genera la predicció $\hat{\mathbf{Y}}_{batch}$ per a la totalitat de la imatge de manera concurrent. El mapa d'errors resultant, $\mathbf{E}_{batch} = \mathbf{G}_{batch} - \hat{\mathbf{Y}}_{batch}$, es restitueix a l'espai original mitjançant la transformació inversa $\mathcal{T}^{-1}$ (combinant `reshape` i `transpose` invers). Finalment, s'aplica el

retall per eliminar el *padding* i restaurar la resolució nativa, assolint la reconstrucció completa en un únic pas computacional.

4.2.5 Optimització en CUDA: Pre-assignació Massiva

L'arquitectura CUDA utilitzada és conceptualment idèntica a la descrita a la Secció 4.1.6, emprant `@cp.fuse` per a les funcions d'activació. La diferència rau en l'escala: el *Parallel* eleva la magnitud de càlcul a N blocs simultanis. Per evitar el *GPU Starvation*, s'ha implementat una política estricta de *Static Memory Allocation*. Tots els tensors (`w_buf`, `b_buf`, `er_p_buf`) es pre-assignen amb `cp.empty` a l'inici, reutilitzant els mateixos punters de memòria de vídeo en cada iteració del *Feedback Loop*. Això elimina completament la latència associada a la gestió de memòria del *runtime* de CUDA, permetent que l'execució se centri exclusivament en l'operativa de coma flotant.

5 AVALUACIÓ EXPERIMENTAL

L'entorn experimental d'aquest estudi s'ha dissenyat per avaluar de manera rigorosa i determinista tant el rendiment computacional com la fidelitat de reconstrucció de les diferents implementacions de l'algorisme proposat (Seqüencial, *Wavefront* i *Parallel*, tant en CPU com en GPU). Totes les operacions s'han executat utilitzant una precisió de coma flotant de 64 bits (Float64) per garantir la màxima fidelitat respecte l'article original.

5.1 Especificacions de l'Entorn

L'entorn s'ha basat en una arquitectura AMD Ryzen 5 5600X, 16 GB de RAM DDR4-3200 MHz. La unitat d'acceleració ha estat una NVIDIA GeForce RTX 3060, la qual ha permès validar l'eficiència dels *Streaming Multiprocessors* (SM) davant les estratègies proposades. La pila tecnològica es compon de Python 3.12.9, *CUDA Toolkit* 12.4.0 (driver 551.61) i *cupy-cuda12x*.

5.2 Conjunt de Dades i Espai de Paràmetres

Les proves s'han dut a terme sobre un conjunt heterogeni de 18 imatges [3], estructurades en classes (A-E) i imatges mèdiques de referència, cobrint un ampli rang de resolucions (des de 416×240 fins a 2560×2048 píxels). Per a garantir una exploració exhaustiva de l'algorisme, s'ha avaluat el producte cartesià dels següents hiperparàmetres:

- **Mida de bloc (B):** $\{4, 8, 16, 32, 64, 128, 256\}$.
- **Taxa d'aprenentatge (α):** $\{0.05, 0.1, 0.5, 0.7\}$.
- **Mida del vector de context (K):** $\{7, 15, 20, 31\}$.

5.3 Protocol d'Execució i Mesura

Per a mitigar els biaixos sistemàtics introduïts per l'assignació inicial de memòria i la compilació en temps d'execució (*Just-In-Time*, JIT) inherent als entorns d'acceleració per maquinari (com CUDA i CuPy), s'ha establert un protocol d'execució estricta en dues fases:

1. **Fase d'escalfament (Warm-up):** Prèviament a la cronometria, cada configuració realitza una iteració inicial de processament sobre un segment reduït de la imatge. Això força la inicialització del context a la GPU i estabilitza els busos de dades.
2. **Fase d'avaluació:** Un cop estabilitzat el sistema, el procés complet de compressió i descompressió s'executa cinc vegades de manera consecutiva.

Els temps d'execució reportats en aquest estudi corresponen a la mitjana aritmètica d'aquestes cinc iteracions, garantint així l'estabilitat estadística de les dades enfront de possibles fluctuacions transitòries del sistema operatiu.

5.4 Mètriques d'Avaluació

El rendiment computacional s'ha quantificat mitjançant la capacitat de processament (*Throughput*, mesurat en megapíxels per segon) i l'acceleració relativa (*Speedup*), utilitzant com a línia base (*baseline*) la implementació seqüencial original de la R2-Net. Paral·lelament, la distorsió de la imatge reconstruïda s'ha avaluat de manera objectiva aplicant mètriques d'error estàndard: l'Error Absolut Pic (PAE), l'Error Quadràtic Mitjà (MSE) i la Relació Senyal-Soroll de Pic (PSNR). Addicionalment, durant l'execució de les proves, un procés de perfilatge en segon pla ha monitorat contínuament la telemetria del maquinari (utilització de CPU, GPU, RAM i VRAM) mitjançant un mostreig a intervals de 100 ms (vegeu l'Apèndix A.3), permetent correlacionar els temps d'execució amb la saturació real dels recursos del sistema.

A més de les mètriques de distorsió, s'ha avaluat la compressibilitat teòrica de les dades mesurant-ne l'entropia, expressada en bits per píxel (*bpp*). Per a la imatge original en format enter, s'ha calculat l'entropia de Shannon clàssica d'ordre zero. D'altra banda, per avaluar l'entropia dels mapes d'error en coma flotant (precisió de 32 o 64 bits), s'ha aplicat un procediment basat en l'estàndard IEEE 754 [10]. Aquest mètode tracta les dades crues a nivell de memòria, descomponent-les en plans de bits (*bitplanes*). Per a cada pla, se'n calcula l'entropia condicional espacial, la qual mesura la incertesa d'un bit tenint en compte l'estat del píxel veí adjacent. El cost d'emmagatzematge teòric total s'obté mitjançant la suma de les entropies condicionals espacials de tots els plans de bits.

5.5 Resultats

A continuació es presenten els resultats empírics obtinguts a partir de l'execució del banc de proves prèviament descrit [4]. Les gràfiques següents comparen el rendiment computacional de les diferents implementacions (Seqüencial, Wavefront i Paral·lel en CPU i GPU) en termes de capacitat de processament i acceleració relativa. Així mateix, s'exposen les mètriques de fidelitat de les imatges reconstruïdes i el perfilatge de l'ús dels recursos del maquinari, permetent una anàlisi exhaustiva de l'eficiència de cada enfocament.

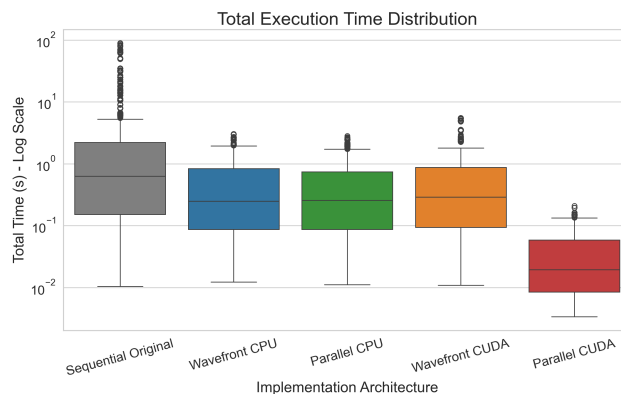


Fig. 7: **Temps d'execució en escala logarítmica.** Aquesta gràfica evidencia la reducció dràstica del temps de processament aconseguida per les implementacions paral·leles en comparació amb l'algorisme seqüencial de referència. L'ús de l'escala logarítmica permet apreciar com la vectorització del codi elimina el bloqueig iteratiu, reduint els temps de resolucions complexes de l'ordre de segons a fraccions de segon.

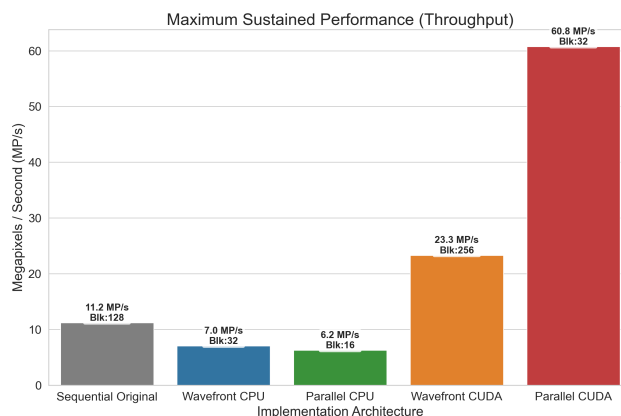


Fig. 8: **Rendiment màxim (Throughput).** Es mesura la quantitat de dades processades en megapíxels per segon (MP/s). Destaca la solidesa de la versió `Parallel CPU`, que manté un flux de dades sostingut i excepcionalment alt per a mides de bloc tancades, posant de manifest l'eficiència de la CPU per tractar estructures de dades que no requereixen l'enviament a través del bus PCIe.

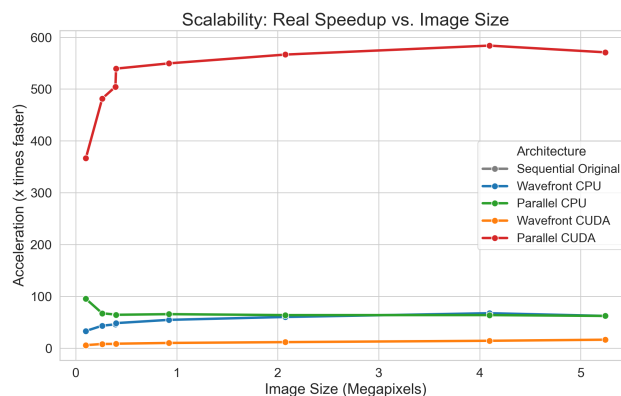


Fig. 9: **Escalabilitat i acceleració (Speedup).** La visualització mostra com evoluciona el guany de rendiment respecte a l'algorisme base a mesura que incrementa la resolució de la imatge. Demostra que l'arquitectura proposada escala de forma robusta, aprofitant al màxim els nuclis disponibles (tant en CPU com en GPU) a mesura que la càrrega de treball s'intensifica.

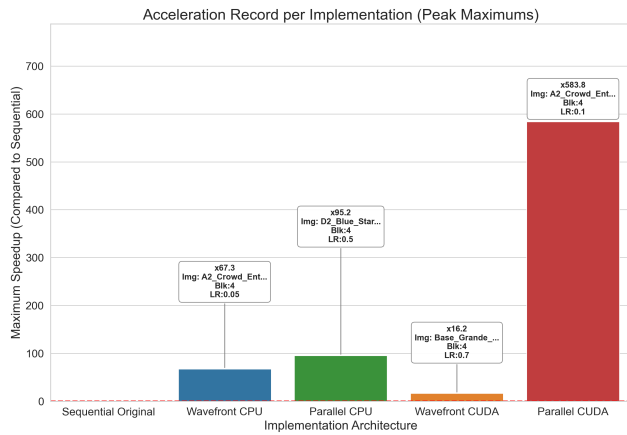


Fig. 10: **Millor configuració global.** Aquesta representació sintetitza les variables de l'estudi per identificar la configuració òptima del sistema. Avalua el balanç ideal entre els temps d'execució, el *Throughput* i el consum de recursos, establint l'estàndard de referència sota el qual la xarxa DPCM assoleix la seva màxima eficàcia teòrica i pràctica.

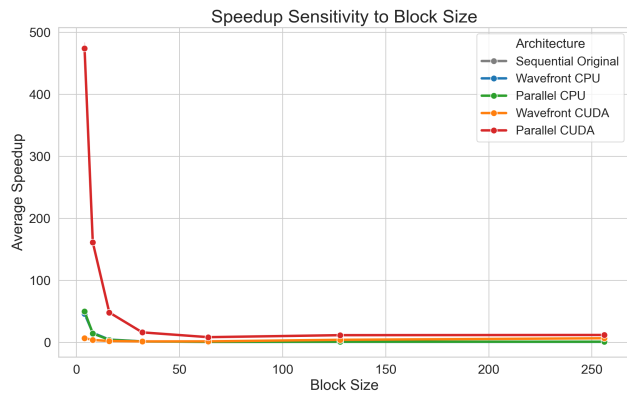


Fig. 11: **Impacte de la mida del bloc.** Analitza el comportament de les diferents arquitectures en variar la mida de la partició (de 4 a 256 píxels). S'observa una correlació directa: mides de bloc petites afavoreixen la latència nul·la de la CPU, mentre que mides més grans són imprescindibles per alimentar els nuclis de la GPU i amortitzar el cost de transferència de memòria.

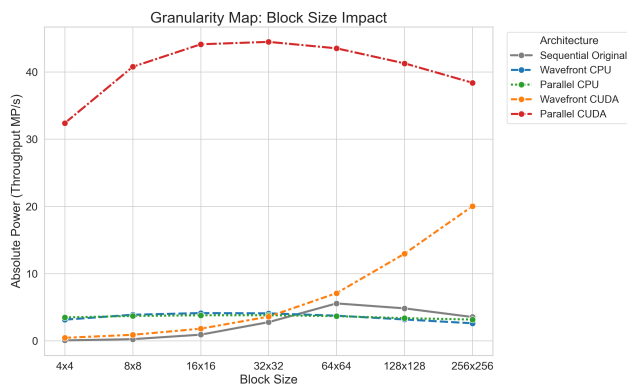


Fig. 12: **Punt de creuament (Full Crossover).** Representa una de les troballes més significatives de l'experimentació. Mostra el llindar exacte de càrrega de treball on les línies de rendiment s'encreuen i el paral·lelisme massiu de CUDA assumeix el lideratge absolut, superant la capacitat de la CPU i justificant l'ús d'acceleració gràfica en altes resolucions.

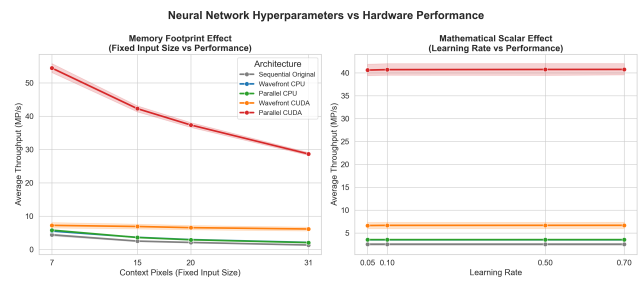


Fig. 13: **Impacte dels hiperparàmetres i l'error.** Il·lustra la relació entre variables com la taxa d'aprenentatge i la minimització de l'error. Confirma que els ajustos necessaris per optimitzar la qualitat de la imatge reconstruïda no afegeixen penalitzacions temporals al flux d'execució paral·lel de l'algorisme.

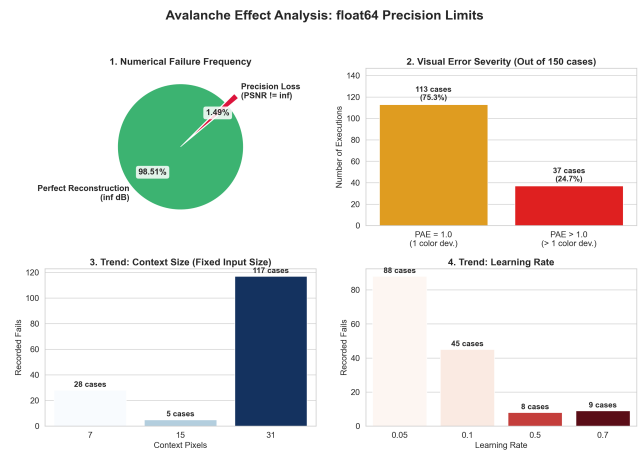


Fig. 14: **Precisió simple vs. doble (float32 / float64).** Aquesta infografia resumeix els beneficis d'optar per estructures de 32 bits. Les dades certifiquen que la precisió float32 duplica pràcticament la velocitat de còmput i redueix a la meitat l'espai en memòria, tot preservant uns valors de qualitat (PSNR i MSE) de forma general indetectable per a l'ull humà.

TAULA 1: Avaluació de la compressibilitat teòrica (en bpp) per a diverses imatges de prova. Es contrasta l'entropia de Shannon de la imatge original (format enter) amb l'entropia condicional espacial acumulada dels mapes d'error produïts pels mètodes d'execució Seqüencial, Wavefront i Paral·lel, operant en precisió de 64 bits (f64) i 32 bits (f32).

Imatge	Original	Seqüencial		Wavefront		Paral·lel	
		f64	f32	f64	f32	f64	f32
A1_2560x1600	7.6685	43.5856	14.5409	43.7067	14.5972	48.6112	21.2387
B2_1920x1080	7.7860	41.0503	12.2774	41.3948	12.5934	48.0588	20.0993
C3_832x480	7.5763	43.4126	14.9006	43.4490	14.9752	50.0498	22.0524

Com es detalla a la Taula 1, l'ús de float64 dispara l'entropia (41-43 bpp) respecte a float32 (12-14 bpp) pel soroll d'arrodoniment incompressible de la mantissa. Addicionalment, mentre els mètodes Seqüencial i Wavefront assoleixen la mateixa compressibilitat, la versió Paral·lel l'empenjora significativament (fins a 22 bpp). Aquest increment reflecteix la pèrdua de correlació espacial: processar blocs aïlladament genera un mapa d'error impredecible que penalitza la compressió teòrica.



Fig. 15: **Comparativa visual de compressió i descompressió segons la precisió de màquina.** A l'esquerra es mostra la imatge original de referència. A la dreta, la fila superior exposa els mapes d'error extrets per l'algorisme, i la inferior presenta els resultats reconstruïts. Com s'observa en les mètriques, l'ús de `float64` aconsegueix una reconstrucció matemàticament perfecta (PAE: 0, MSE: 0, PSNR: ∞). D'altra banda, la precisió `float32` introdueix una pèrdua d'informació completament imperceptible per a l'ull humà, reportant un Error Absolut Pic (PAE) de 1.0000, un Error Quadràtic Mitjà (MSE) de 0.0017 i una Relació Senyal-Soroll de Pic (PSNR) de 75.87 dB.



Fig. 16: **Comparativa visual dels mapes d'error segons el paradigma d'execució.** S'observen els residus generats per les versions Seqüencial, Wavefront i Paral·lel operant sota la mateixa configuració de paràmetres. Com es pot apreciar, els mapes de les versions Seqüencial i Wavefront resulten visualment idèntics, atès que ambdós paradigmes respecten les dependències de dades causals, preservant així la continuïtat espacial de la imatge. En fort contrast, el mapa d'error de la versió Paral·lel mostra una aparença molt més plana i grisa, juntament amb l'aparició de l'efecte graella altament evidents. Aquesta divergència es deu al fet que el processament simultani de tots els blocs impedeix la retroalimentació del context realment reconstruït entre blocs veïns, trencant la correlació espacial i generant discontinuïtats a les vores.

6 CONCLUSIONS

A partir de l'anàlisi de les dades experimentals, es poden extreure les següents conclusions sobre l'acceleració de l'algorisme R2-Net:

I) Resultats de rendiment i acceleració GPU: L'arquitectura CUDA ha permès assolir un *speedup* màxim de **583.8x** respecte a l'execució seqüencial en CPU (Fig. 9). Aquesta xifra valida el potencial de la GPU per a vectorització massiva i confirma el punt de creuament on l'acceleració gràfica supera clarament la CPU en altes resolucions.

II) Limitacions del mètode Wavefront: L'anàlisi ha demostrat que aquest esquema, tot i resoldre les dependències causals, presenta una escalabilitat limitada per la latència del *runtime* de Python en el llançament de les operacions de CUDA (Fig. 11). Motiu pel qual el mètode *Parallel* esdevé computacionalment superior en grans volums de dades.

III) Equilibri en la precisió numèrica: L'ús de la precisió `float32` és l'opció més equilibrada (Fig. 14). Permet duplicar la velocitat de còmput i reduir a la meitat el consum de memòria respecte a `float64`, mantenint una qualitat de reconstrucció amb errors imperceptibles (PSNR de 75.87 dB), tal com es pot apreciar a la Fig. 15.

IV) Compressibilitat (Entropia): S'ha demostrat que prioritzar la velocitat absoluta té un cost directe sobre la compressió (Taula 1). En trencar la correlació espacial per processar els blocs aïlladament, el soroll d'arrodoniment pur de la mantissa esdevé aleatori i incompressible, penalitzant severament la taxa teòrica de bits respecte a les variants seqüencials, com evidencien els mapes d'error de la Fig. 16.

Treball Futur i Millores

Durant la fase final d'aquest projecte s'han identificat i explorat tècniques d'optimització addicionals que, a causa de limitacions temporals, no s'han pogut integrar de forma exhaustiva al banc de proves principal, però que perfilen unes línies de recerca futures altament prometedores per reduir la taxa de bits:

I) Integració del format `float16` i quantització adaptativa S'ha observat que aplicar una quantització just abans de la predicció de la xarxa, i posteriorment utilitzar aquesta mateixa quantització per emmagatzemar l'error residual en `float16`, permet mantenir la potència del càlcul intern en `float32` o `float64` sense deteriorar el PSNR, reduint dràsticament l'espai d'emmagatzematge.

II) Quantització selectiva de la mantissa (IEEE 754): En aprofitar l'estructura binària del punt flotant estàndard [10] (signe, exponent i mantissa), s'ha experimentat amb la quantització exclusiva dels bits menys significatius de la mantissa, fixant-los a zero mitjançant un pas (*qstep*) en potències de 2. Això elimina de soca-rel el soroll numèric latent. Proves preliminars demostren que, aplicant un *qstep* = 16, l'entropia condicional espacial es desploma fins a 6.03 bpp, mantenint un PAE de 4.0 i un PSNR excepcionalment alt de 52.37 dB. A més, augmentant l'agressivitat fins a un *qstep* = 256, s'ha assolit una entropia de només 3.19 bpp (molt per sota dels 7.78 bpp de la imatge original de referència). Tot i registrar un PSNR de 28.59 dB en aquest darrer escenari, les diferències visuals resulten sorprenentment subtils depenent de la textura de la imatge original.

ÚS DE LA IA GENERATIVA

L'ús de la intel·ligència artificial s'ha limitat exclusivament a funcions de suport en la redacció i refinament de la terminologia i connotació matemàtica.

REFERÈNCIES

- [1] V. Sanchez, "Overfitted Neural Networks for Block-based Intra-prediction", *Data Compression Conference (DCC)*, 2024. [Online]. Available: <https://ieeexplore.ieee.org/document/10533747>
- [2] CuPy Development Team, "CuPy: A NumPy-compatible array library accelerated by CUDA". [Online]. Available: <https://cupy.dev/>
- [3] A. Blesa Rodríguez, "R2Net custom dataset". Zenodo, May 18, 2026. doi: 10.5281/zenodo.20268850. [Online]. Available: <https://zenodo.org/records/20268850>
- [4] A. Blesa Rodríguez, "R2Net Computational Benchmark and Hardware Performance Datasets (Float32 vs. Float64)". Zenodo, May 22, 2026. doi: 10.5281/zenodo.20347920. [Online]. Available: <https://zenodo.org/records/20347920>
- [5] Stanford Vision and Learning Lab, "Convolutional Neural Networks: Spatial Arrangement and Padding", *CS231n: Deep Learning for Computer Vision*. [Online]. Available: <https://cs231n.github.io/convolutional-networks/>
- [6] NumPy Developers, "numpy.matmul (Batch Matrix Multiplication)", *NumPy Reference Documentation*. [Online]. Available: <https://numpy.org/doc/stable/reference/generated/numpy.matmul.html>
- [7] M. Snir, "Wavefront Parallel Pattern", *CS 533 Parallel Computer Architecture*, University of Illinois at Urbana-Champaign. [Online]. Available: <https://snir.cs.illinois.edu/patterns/wavefront.pdf>
- [8] CuPy Development Team, "User-Defined Kernels: cupy.fuse", *CuPy Documentation*. [Online]. Available: <https://docs.cupy.dev/en/stable/reference/generated/cupy.fuse.html>
- [9] NumPy Developers, "Broadcasting Rules on Multidimensional Arrays", *NumPy User Guide*. [Online]. Available: <https://numpy.org/doc/stable/user/basics.broadcasting.html>
- [10] IEEE Computer Society, "IEEE Standard for Floating-Point Arithmetic (IEEE 754)", *IEEE Xplore*, 2019. [Online]. Available: <https://ieeexplore.ieee.org/document/8766229>
- [11] NVIDIA Corporation, "NVIDIA Ampere Architecture Tuning Guide", *CUDA Toolkit Documentation*. [Online]. Available: <https://docs.nvidia.com/cuda/ampere-tuning-guide/index.html>

APÈNDIX

A.1 Extracció Vectoritzada del *Ground Truth*

L'enfocament iteratiu clàssic per extreure les dades d'una imatge consistiria a utilitzar dos bucles niats per recórrer l'alçada i l'amplada de cada bloc de forma individual. Per maximitzar l'eficiència computacional, l'extracció dels píxels originals (el *Ground Truth*) es realitza de forma massiva i simultània per a tots els N blocs de la diagonal activa mitjançant tècniques d'indexació avançada i expansió de dimensions (*broadcasting*) [9].

Aquest procés comença amb la determinació de les coordenades d'origen de cada bloc dins la matriu d'imatge original $\mathbf{I}$ (i no sobre la matriu de reconstrucció estesa $\mathbf{R}_{pad}$). Formalment, definim $\mathbf{r}_s \in \mathbb{N}^N$ com el vector que conté els índexs de les files per als blocs actius a la diagonal k :

$$\mathbf{r}_s = [\max(0, k - N_{cols} + 1), \dots, \min(N_{rows} - 1, k)]^T$$

A causa de la geometria de les anti-diagonals, l'índex de les columnes corresponents es calcula mitjançant el vector $\mathbf{c}_s \in \mathbb{N}^N$, el qual es deriva restant $\mathbf{r}_s$ a l'índex de la diagonal:

$$\mathbf{c}_s = k \cdot \mathbf{1} - \mathbf{r}_s$$

Tal com s'il·lustra a la Fig. 5, aquests vectors identifiquen exclusivament la posició matricial discreta. Per traslladar aquest comportament al domini físic de la imatge, les coordenades absolutes en píxels del vèrtex d'ancoratge (la cantonada superior esquerra) s'obtenen escalant els índexs matricials per la mida del bloc b , definint així els vectors d'origen espacial $\mathbf{p}_y$ i $\mathbf{p}_x$:

$$\mathbf{p}_y = b \cdot \mathbf{r}_s$$

$$\mathbf{p}_x = b \cdot \mathbf{c}_s$$

Atès que aquests dos vectors de longitud N actuen merament com a punters cap a l'inici geomètric de cada regió, el repte computacional recau en transformar aquests N punts singulars en un tensor tridimensional que contingui la totalitat dels píxels que componen l'àrea interior de cada bloc. Per estendre la informació des d'aquestes cantonades singulars cap a l'àrea bidimensional de la matriu de píxels, es defineix un vector de desplaçaments locals $\Delta = [0, 1, \dots, b-1]^T$. L'objectiu és sumar aquests desplaçaments a les coordenades d'origen per «dibuixar» la quadrícula completa de cada bloc.

No obstant això, per generar coordenades bidimensionals, l'expansió de les files i de les columnes s'ha de realitzar en direccions ortogonals mitjançant la injecció estratègica de dimensions artificials (*broadcasting*, implementat amb l'operador `None` al codi). D'una banda, per generar la graella de les coordenades verticals $\mathbf{R}_{grid}$, el vector de desplaçaments s'orienta en el segon eix (forma $1 \times b \times 1$). En sumar-lo als orígens $\mathbf{p}_y$ (forma $N \times 1 \times 1$), l'operació s'expandeix generant un tensor de dimensions $N \times b \times 1$:

$$\mathbf{R}_{grid}(n, u, 1) = \mathbf{p}_y(n) + \Delta(u)$$

D'altra banda, per a la graella de coordenades horitzontals $\mathbf{C}_{grid}$, el vector de desplaçaments s'ha d'orientar en el

tercer eix (forma $1 \times 1 \times b$). En sumar-lo a $\mathbf{p}_x$, s'expandeix en la direcció horitzontal:

$$\mathbf{C}_{grid}(n, 1, v) = \mathbf{p}_x(n) + \Delta(v)$$

on $n \in \{0, \dots, N-1\}$ indexa cada bloc de la diagonal, u representa l'increment vertical i v l'increment horitzontal. El motor de computació, en rebre aquests dos tensors de coordenades, els expandeix virtualment fins a conformar una graella completa de dimensions $N \times b \times b$.

Per visualitzar la mecànica computacional d'aquesta operació en paral·lel, considerem un escenari amb una mida de bloc $b = 2$ i una diagonal activa que conté $N = 2$ blocs independents. Suposem que els orígens verticals d'aquests dos blocs són $\mathbf{p}_y = [0, 2]^T$ i el vector de desplaçaments locals és $\Delta = [0, 1]^T$. A l'avaluar l'equació per a $\mathbf{R}_{grid}$, el motor matemàtic processa ambdós blocs simultàniament al llarg de l'eix N , suma el desplaçament a les files, i **clona automàticament els valors** al llarg de la dimensió de les columnes per emplenar l'espai buit. El resultat és un únic tensor tridimensional:

$$\mathbf{R}_{grid} = \begin{bmatrix} \begin{bmatrix} 0+0 & 0+0 \\ 0+1 & 0+1 \end{bmatrix} \\ \begin{bmatrix} 2+0 & 2+0 \\ 2+1 & 2+1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix} \\ \begin{bmatrix} 2 & 2 \\ 3 & 3 \end{bmatrix} \end{bmatrix}$$

Aquest tensor multidimensional indica que, per reconstruir la primera fila del primer bloc ($n = 0$), s'ha de llegir la fila absoluta 0 de la imatge dues vegades; mentre que per al segon bloc ($n = 1$), s'ha de llegir la fila absoluta 2. Al combinar ortogonalment aquesta informació amb el tensor $\mathbf{C}_{grid}$, s'obtenen les parelles de coordenades exactes per a tots els píxels de tots els blocs de la diagonal.

En avaluar la imatge estesa en aquestes coordenades tridimensionals de forma concurrent, $\mathbf{I}_{pad}[\mathbf{R}_{grid}, \mathbf{C}_{grid}]$, s'extreuen totes les regions d'interès en una única instrucció de memòria. Finalment, atès que l'arquitectura del perceptró multicapa de la R2Net requereix que l'entrada de cada bloc s'estructuri com un vector columna, aquest tensor resultant (de forma $N \times b \times b$) s'aplanarà en les seves dimensions espacials. El tensor definitiu de *Ground Truth*, $\mathbf{G}^{(k)} \in \mathbb{R}^{N \times b^2 \times 1}$, unifica així la informació de tota la diagonal, quedant preparat per ser processat per la xarxa neuronal en una sola operació matricial.

A.2 Reducció Dimensional i Mapeig del Context (Matriu $\mathbf{M}$)

L'arquitectura de la xarxa neuronal requereix una capa d'entrada de mida fixa, denotada com K (`FIXED_INPUT_SIZE`). No obstant això, el context espacial en forma de « $\mathbf{L}$ » invertida extret al voltant d'un bloc de mida $b \times b$ està format per b píxels de la vora esquerra, 1 píxel de la cantonada superior-esquerra i b píxels de la vora superior. Això resulta en un vector de context de longitud variable $L = 2b + 1$ per al Wavefront ($L = 2b - 1$ per al mètode Parallel).

Atès que L i K rarament coincideixen (i sovint $L > K$), és necessari aplicar una operació de *downsampling* tal com fa la R2Net original (però vectoritzada) per mapar el vector original $\mathbf{x} \in \mathbb{R}^L$ cap al vector d'entrada de la xarxa

$\mathbf{x}' \in \mathbb{R}^K$. Per realitzar aquesta operació maximitzant el rendiment i evitant l'ús de bucles iteratius durant l'execució del *Wavefront*, es precalcu la matriu de transformació o projecció $\mathbf{M} \in \mathbb{R}^{L \times K}$.

L'estratègia consisteix a dividir els L píxels en K subgrups contigus de manera quasi-equitativa. Si definim S_k com el conjunt d'índexs dels píxels que pertanyen al k -èsim grup, i $s_k = |S_k|$ com la seva cardinalitat, els coeficients de la matriu $\mathbf{M}$ es defineixen com:

$$\mathbf{M}_{l,k} = \begin{cases} \frac{1}{s_k} & \text{si } l \in S_k \\ 0 & \text{en cas contrari} \end{cases}$$

Aquesta formulació converteix l'operació d'agrupació per mitjana en una simple multiplicació matricial. Així, el vector d'entrada per a la xarxa es pot obtenir de manera altament paral·lelitzada mitjançant el producte:

$$\mathbf{x}' = \mathbf{x} \cdot \mathbf{M}$$

Per il·lustrar aquest procés, considerem un exemple on el context extret té $L = 5$ píxels i la xarxa requereix una entrada de $K = 3$. L'algorisme de divisió agrupa els elements de la manera més equitativa possible, resultant en mides de grup $s_1 = 2$, $s_2 = 2$ i $s_3 = 1$. La matriu de projecció $\mathbf{M} \in \mathbb{R}^{5 \times 3}$ resultant s'estructura de la següent manera:

$$\mathbf{M} = \begin{pmatrix} 0.5 & 0 & 0 \\ 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Si multipliquem el nostre vector de context original $\mathbf{x} = (x_1 \ x_2 \ x_3 \ x_4 \ x_5)$ per aquesta matriu, obtenim directament el vector submostrejat $\mathbf{x}' \in \mathbb{R}^3$, on cada element és la mitjana exacta del seu grup:

$$\mathbf{x}' = \mathbf{x} \cdot \mathbf{M} = \left(\frac{x_1 + x_2}{2} \quad \frac{x_3 + x_4}{2} \quad x_5 \right)$$

Aquest exemple demostra com l'ús de la matriu $\mathbf{M}$ substitueix completament la necessitat d'iterar manualment sobre els píxels, delegant el càlcul a l'aritmètica de matrius optimitzada per hardware. Addicionalment, la funció de configuració genera un vector booleà de validesa, $\mathbf{v} \in \{0,1\}^K$ (*valid_groups*). Aquesta variable gestiona el cas extrem on la mida d'entrada de la xarxa sigui més gran que el nombre de píxels disponibles ($K > L$). En aquest escenari, algunes columnes de $\mathbf{M}$ quedarien buides ($s_k = 0$). El vector $\mathbf{v}$ actua com una màscara que permet identificar ràpidament aquestes agrupacions nul·les per assignar-los a posteriori el valor neutre de 0.5, garantint així que la xarxa neuronal no rebí valors indefinits ni pèrdues de gradient durant la inferència.

A.3 Monitoratge de pics de maquinari

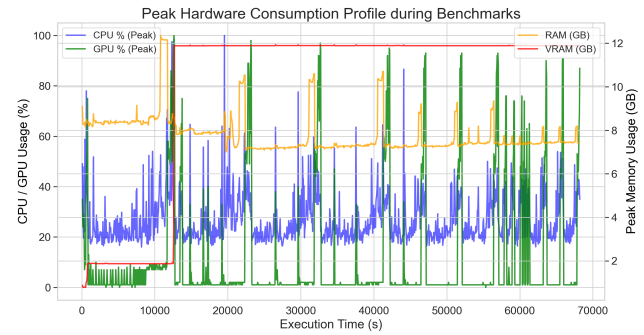


Fig. 17: S'analitza l'impacte computacional sobre els recursos del sistema durant les execucions. L'increment en l'ús de RAM i VRAM és proporcional a la mida del bloc d'entrada, però es manté en tot moment dins de llindars segurs i acotats. Això certifica l'èxit de les tècniques d'optimització, com la preassignació de memòria (*zero-allocation*), que eviten el desbordament i la saturació de la targeta gràfica.