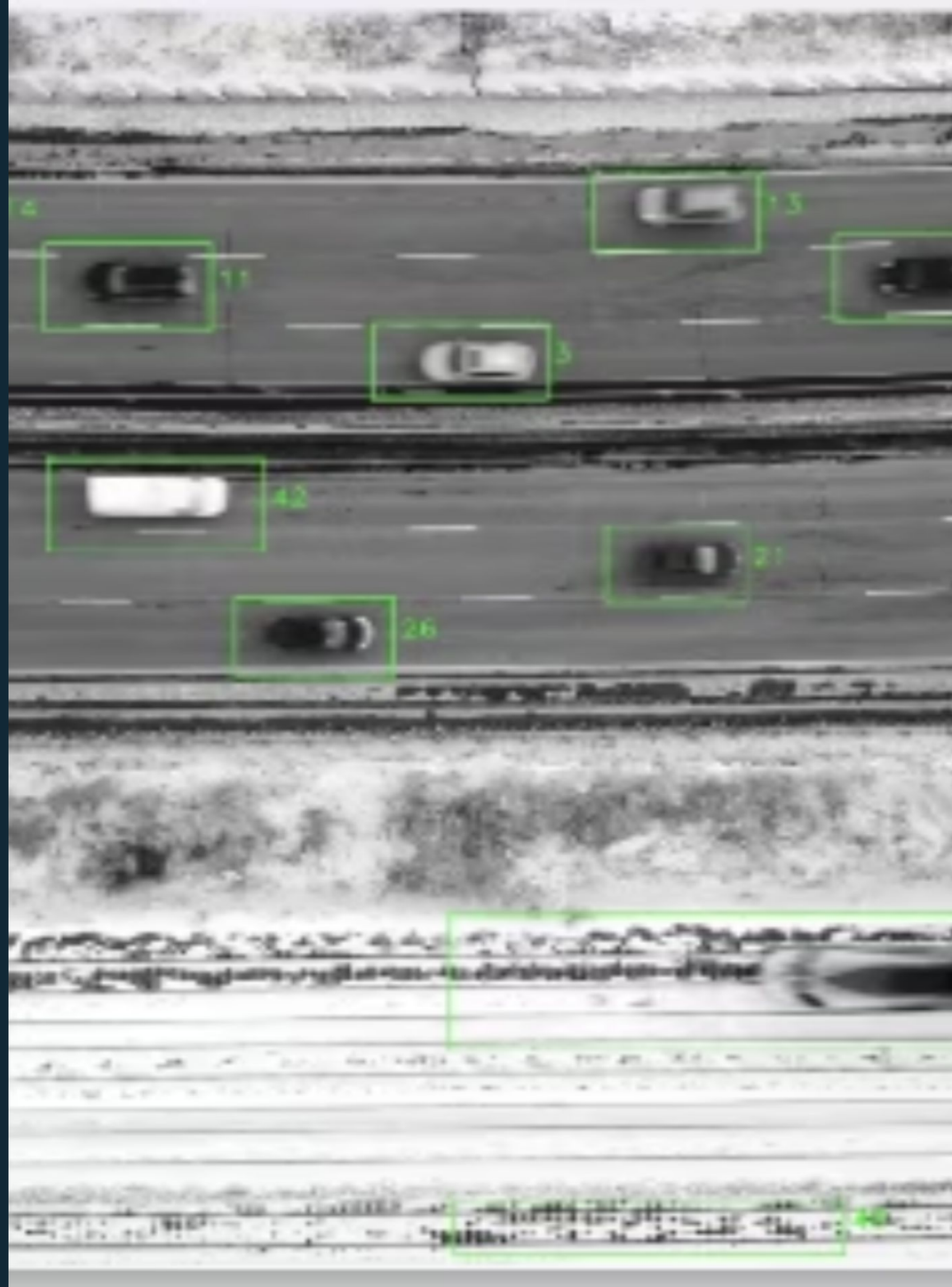
Projet Motion:

Optimisation de la Détection et du Suivi d'Objets en Mouvement

Réalisé par:
Papa Talla Dioum



Baseline et Point de Départ : Motion2

Fig 1: Graphe de la détection de mouvement et du suivi.

Performance Initiale

motion2 : 59.102 ms par image → 16.92 FPS

Simplification de Motion2 : Élimination du Traitement Redondant

L'optimisation majeure consiste à ne calculer la chaîne complète que pour l'image t, puis à permuter les pointeurs des buffers pour l'itération suivante. Cela supprime le traitement explicite à t-1

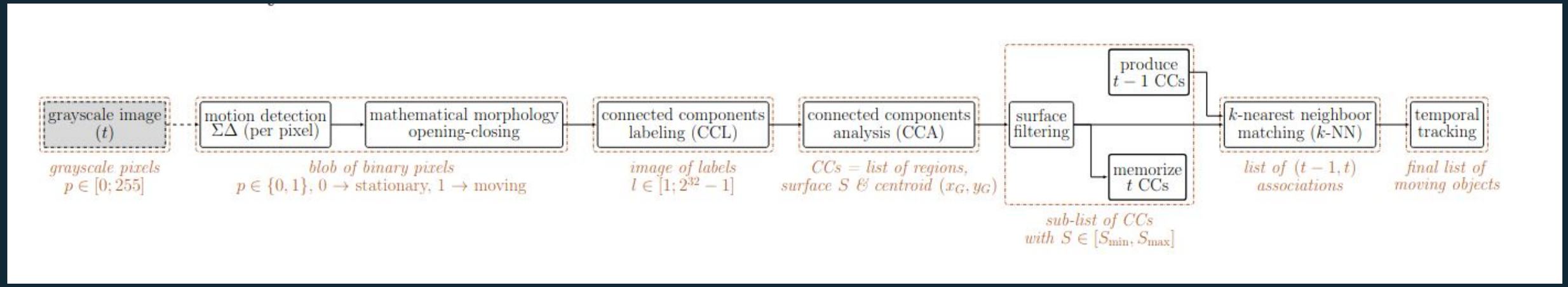


Fig 2 : Graphe de Motion simplifiée.

Performance après Simplification

motion (baseline optimisée) : 29.605 ms → 33.78 FPS

Gain immédiat : x2 par rapport à motion2.

Parallélisation OpenMP du Sigma-Delta

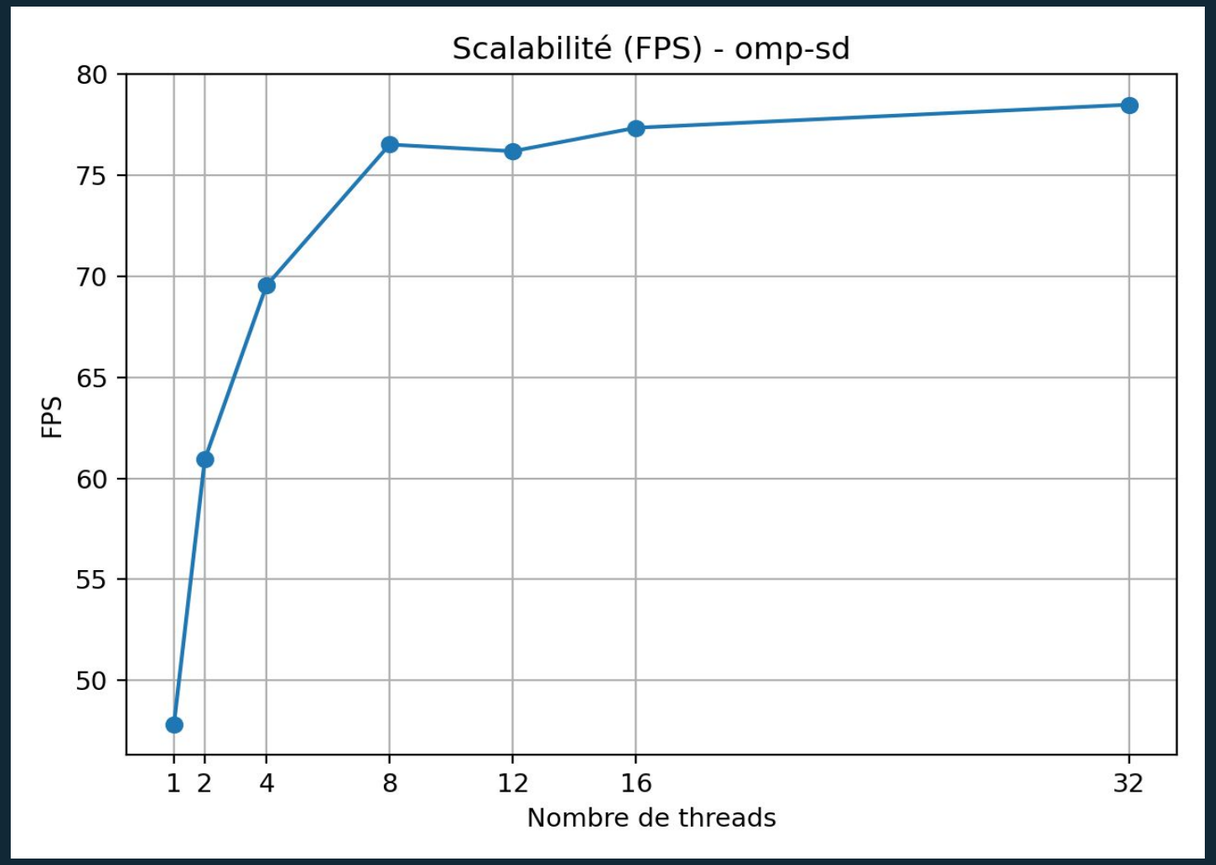
Le Sigma-Delta, identifié comme un goulot d'étranglement, a été parallélisé avec OpenMP pour améliorer les performances.

Les optimisations incluent :

- l'ajout de `#pragma omp for` sur les boucles
- le regroupement de plusieurs boucles dans une région parallèle unique.

omp-sd : 21.840 ms [~45.79 FPS]

omp-sd-v2 : 13.033 ms [~76.73 FPS]



Scalabilité (FPS) - omp-sd

Le nombre de threads a un impact significatif sur les FPS, avec une amélioration notable jusqu'à 32 threads.

Parallélisation OpenMP de la Morphologie

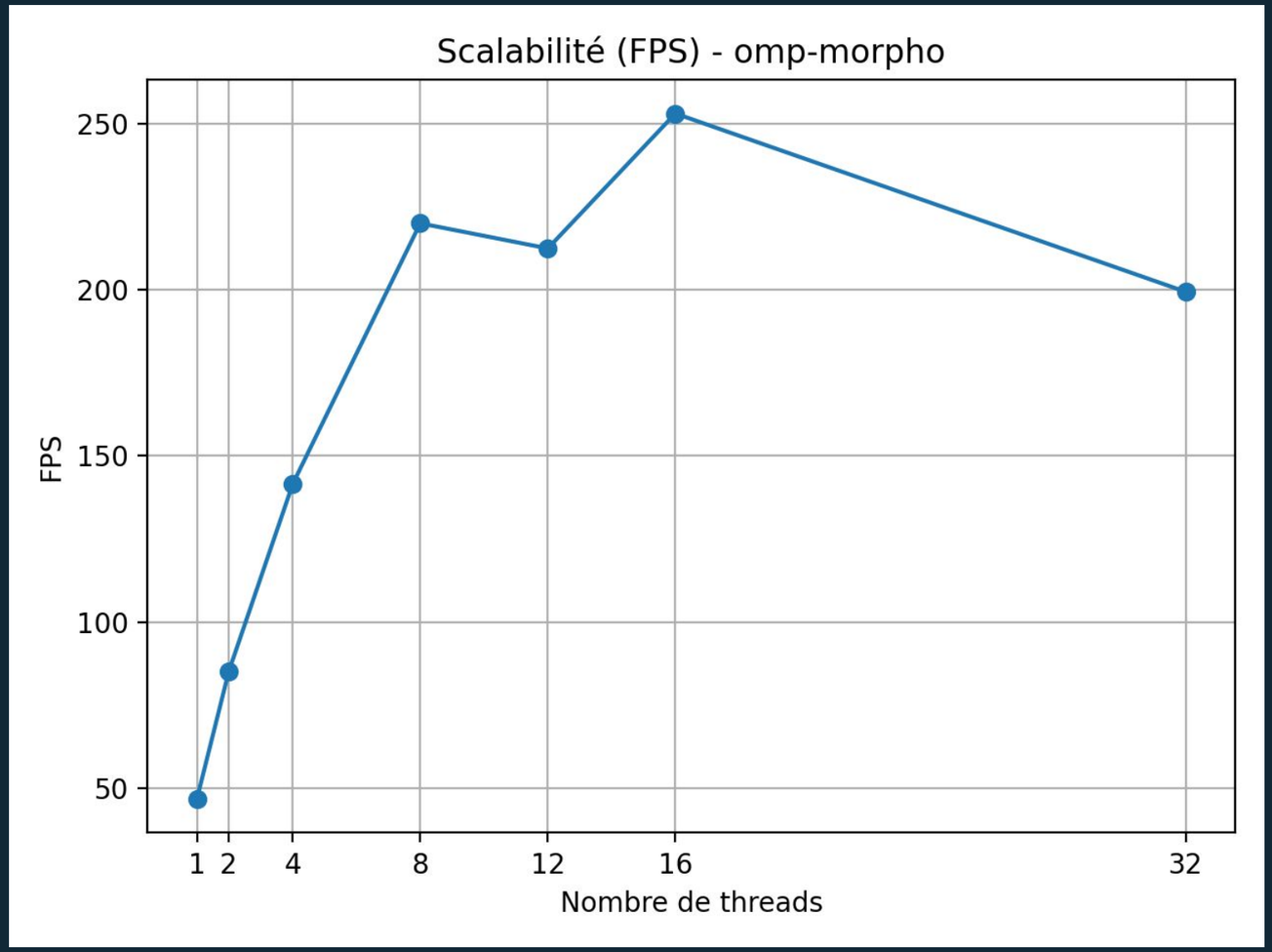
La même méthode de parallélisation OpenMP a été appliquée aux opérateurs morphologiques érosion et dilatation (l'ajout de `#pragma omp for` sur les boucles)
Résultat immédiat : 5.270 ms
[~189.75 FPS]

Amélioration:

- Maintien du calcul sequentiel sur les bords
- Parallélisation ciblée sur le cœur de l'image ont conduit à des gains importants.

Après optimisation ciblée : 224.78 FPS

```
CPU(s):                          20
On-line CPU(s) list:             0-19
Model name:                      13th Gen Intel(R) Core(TM) i9-13900H
Thread(s) per core:              2
Core(s) per socket:              14
CPU(s) scaling MHz:              12%
CPU max MHz:                     5400.0000
CPU min MHz:                     400.0000
NUMA node0 CPU(s):               0-19
```



Scalabilité (FPS) - omp-morpho

Le FPS augmente jusqu'à 16 threads, puis diminue légèrement à 32 threads, indiquant un point d'optimisation.

Optimisation du CCL et CCA

Le Connected Component Labeling (CCL) et le Connected Component Analysis (CCA) ont également bénéficié d'optimisations ciblées.

CCL (Line Segment Labeling)

Algorithme LSL (Line Segment Labeling) :

- Détection de segments ligne par ligne,
- Construction d'équivalences partiellement séquentielle.

Solutions

- Parallélisation complète de la détection de segments.
- Parallélisation de la labellisation finale.
- Réduction des écritures mémoire.

Résultat omp-ccl : 310.14 FPS

CCA (Accumulateurs par Thread)

- Accumulation concurrente sur les RoIs,
- Utilisation d'atomiques → Un seul thread à la fois peut modifier cette variable.

Solutions

- Chaque thread a ses buffers locaux (surface, barycentre, bounding box).
- Aucune opération atomique dans la boucle pixel.
- Fusion finale parallèle.

Résultat CCA : 249.39 FPS

Optimisation Mémoire : Row Pointers

L'utilisation de pointeurs de lignes au lieu d'indices 2D (img[i][j]) pour l'accès aux pixels est une optimisation clé.

Principe

Accès direct aux lignes via des pointeurs (`const uint8_t* r0 = img[i-1];`).

Bénéfices

Réduction des calculs d'adresses, meilleure localité cache, vectorisation facilitée.

Impact Mesuré

sd-rowptr: 390.15 FPS

morpho-rowptr : 555.90 FPS

Cette optimisation s'est avérée être l'une des plus rentables du projet.

Vectorisation SIMD du Sigma-Delta

La vectorisation SIMD (Single Instruction, Multiple Data) a été appliquée au Sigma-Delta, une étape critique, pour exploiter le parallélisme au niveau des données.

Adaptation Idéale

Opérations locales, régulières et indépendantes par pixel, parfaites pour le traitement SIMD.

Principe d'Implémentation

Chargement, calculs et écriture de blocs contigus de pixels en parallèle à l'aide de registres SIMD (AVX2).

Bénéfices Majeurs

Suppression des branchements, maximisation du débit par cycle et exploitation du parallélisme intra-cœur. Le Sigma-Delta devient marginal.

Cette optimisation a permis d'atteindre **702 FPS**

Influence du Nombre de Threads et de l'Allocation CPU

L'allocation explicite des cœurs CPU est cruciale pour maximiser les performances, notamment sur des architectures comme Dalek.

Allocation CPU

Commande `srun --cpus-per-task=16` pour une allocation optimale.

Pic de Performance

Temps total : 0.549 ms

Débit : **≈ 1820 FPS**