IRL

Reconstruction efficace d'un volume 3D isotrope à partir d'images ultrason 2D localisées

Jean-Baptiste Keck
Laboratoire **TIMC-IMAG**
28/05/14

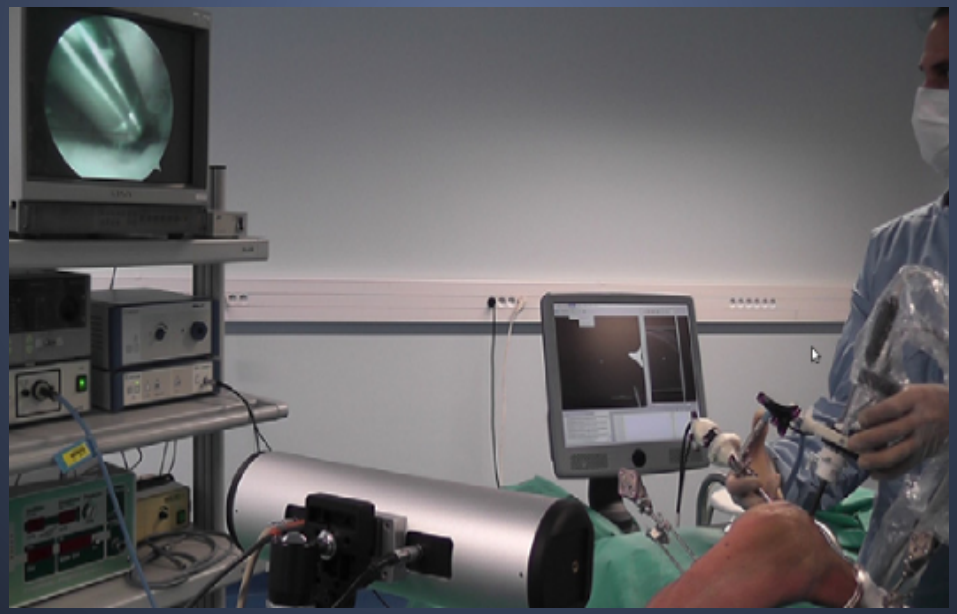
Encadrant : Matthieu Chabanas (TIMC-IMAG équipe GMCAO)
Responsable IRL : Marie-Paule Cani

Plan

1) Introduction et cadre de cet IRL
2) Pourquoi programmer sur GPU ?
3) Reconstruction du volume
4) Choix de l'algorithme et implémentation
5) Résultats obtenus
6) Bilan et conclusion

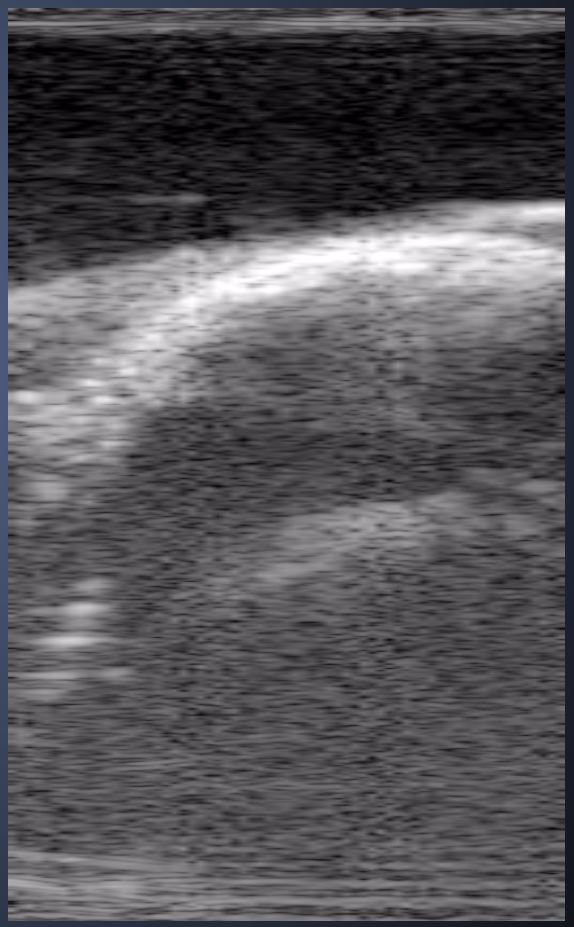
1) Introduction

- L'arthrose, un problème de santé publique
- Développement d'une sonde échographique miniaturisée :



Données générées

- Images
- Positions + orientations (matrices 4x4)



Taille logique des images :
64x1296 (pixels)

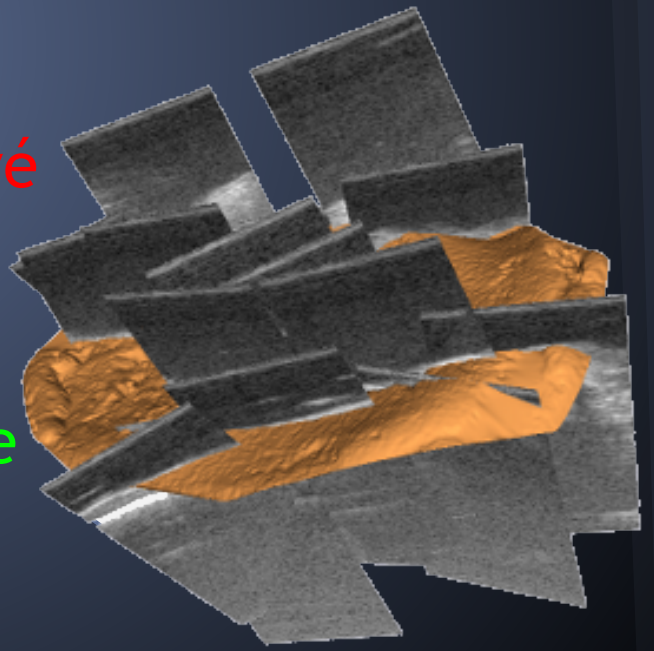
Précision des capteurs :
205x18µm

Taille physique des images :
13x23mm

Nombre d'images générées:
Demi plateau tibial => 1610
Col du fémur => 3770

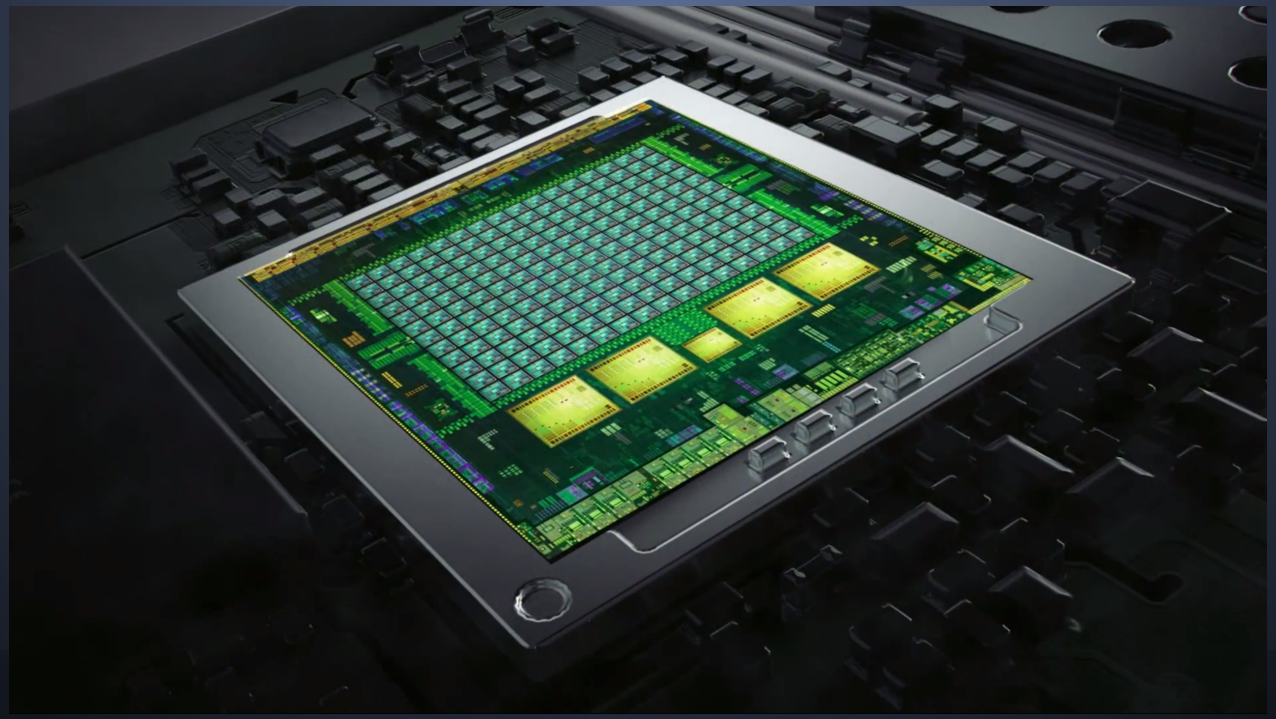
Problèmes et objectifs

- Perception 2D très limitée, pas de vue d'ensemble
- Reconstruction d'un volume pour aider le clinicien dans son diagnostic
- Temps de reconstruction trop élevé pour une application clinique
- Optique de reconstruction efficace sur GPU (Graphic Processing Unit)



2) Pourquoi programmer sur GPU ?

- Grande quantité de processeurs (jusqu'à 5000+ par GPU)
- Paradigme SIMD (Single Instruction Multiple Data)
- Ce type d'utilisation du GPU, le GPGPU, est de plus en plus mature (CUDA, OpenCL)
- Pas encore utilisé pour ce type de reconstruction volumique



3) Reconstruction du volume

Etape 1: Chargement des images et des transformations (matrices 4x4)

Etape 2: Traitement préliminaire (filtrage des images et des transformations, recadrage, conversion)

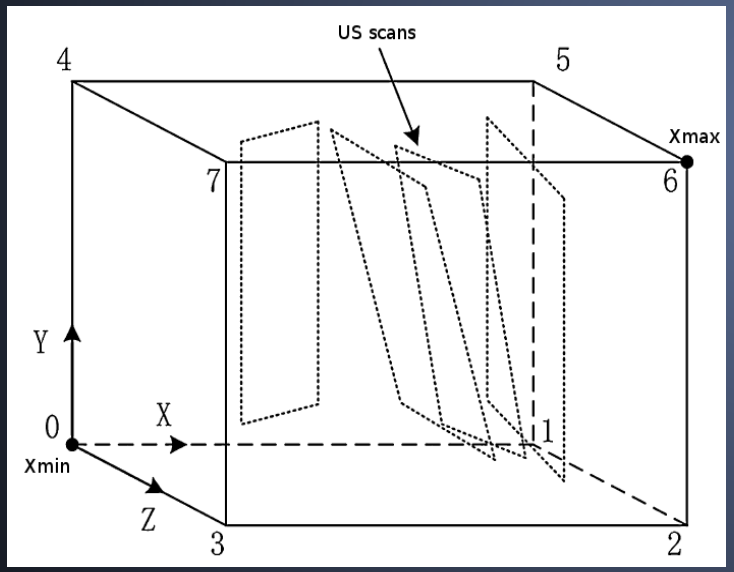
Etape 3: Détermination de la taille de la grille, choix de la taille des voxels et découpe

Etape 4: Reconstruction du volume

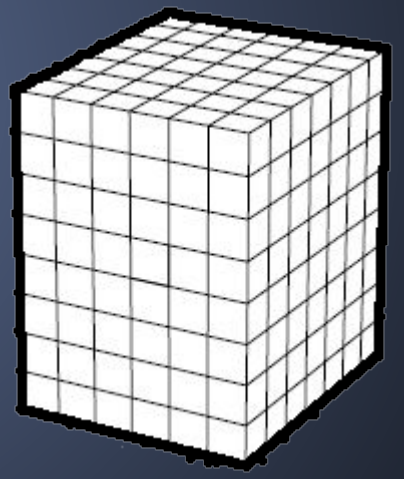
Pour plus d'informations sur ces étapes se référer à T.Wen *et al.* **[1]**

Détermination de la taille de la grille :

1) Determination de la boîte englobante :



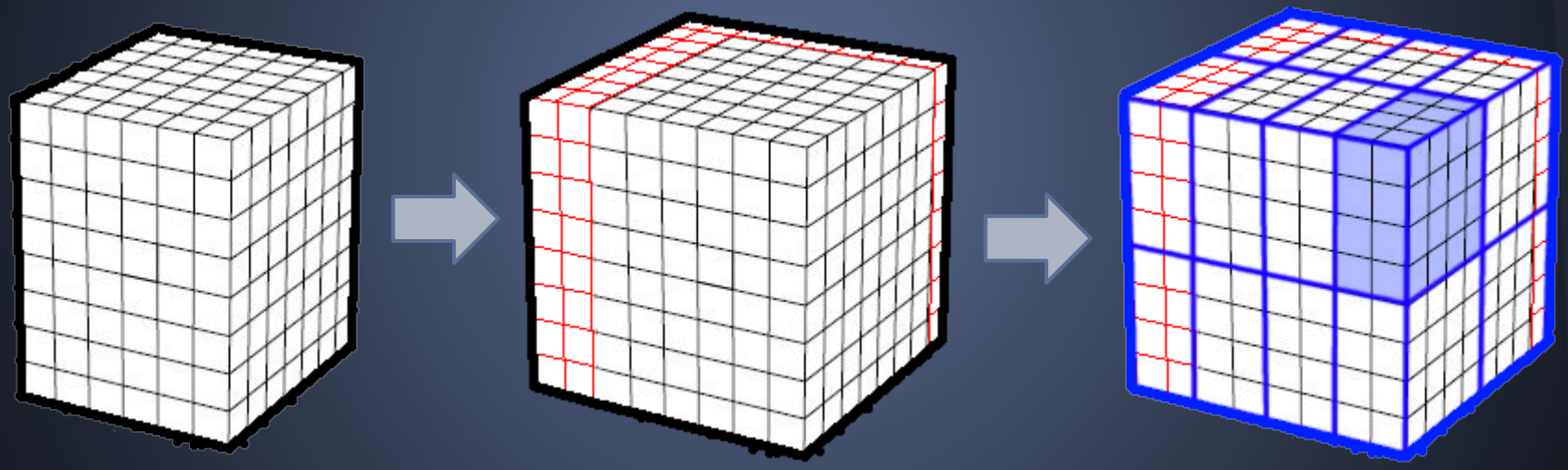
2) Choix de la taille des voxels cubiques :



Choix de la taille des voxels : 0.1 à 0.2mm

Découpage de la grille

- Sur-grille alignée sur une puissance de deux
- Découpe en sous-grilles (contraintes de mémoire) :

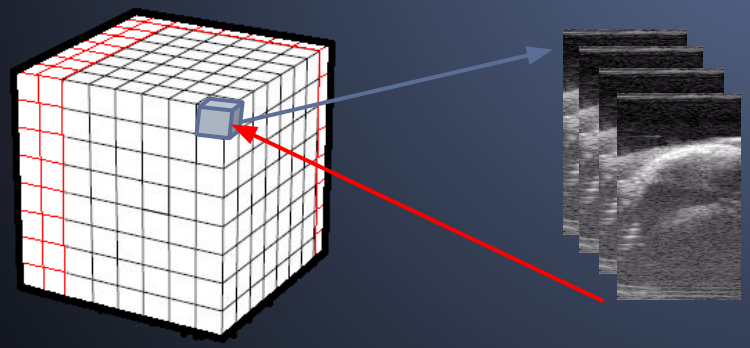


- Permet de mettre en place une solution multi-GPU, d'équilibrer la charge et les transferts de mémoire

Algorithmes de reconstruction :

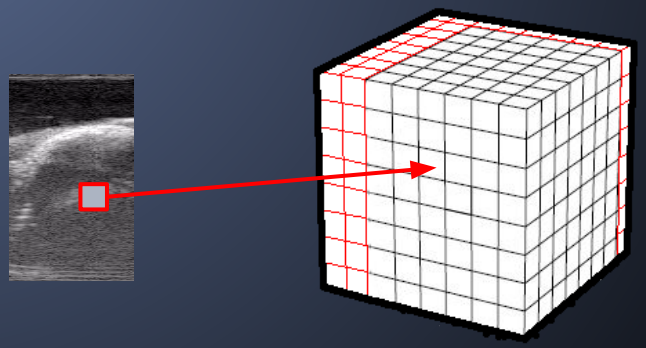
VNN (Voxel Nearest Neighbor)

- Parcours de tous les voxels, puis détermination du pixel le plus proche
- Pas de trous, une seule étape
- Complexité $\mathcal{O}(\mathbf{np})$



PNN (Pixel Nearest Neighbor)

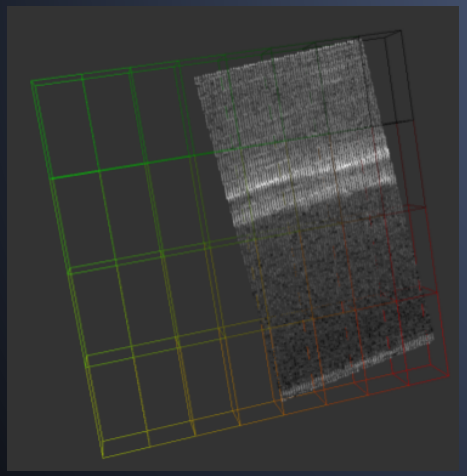
- Parcours de tous les pixels, affectation au voxel le plus proche
- Présence de voxels vides, étape d'interpolation nécéssaire
- Complexité $\mathcal{O}(\mathbf{p + nr^3})$

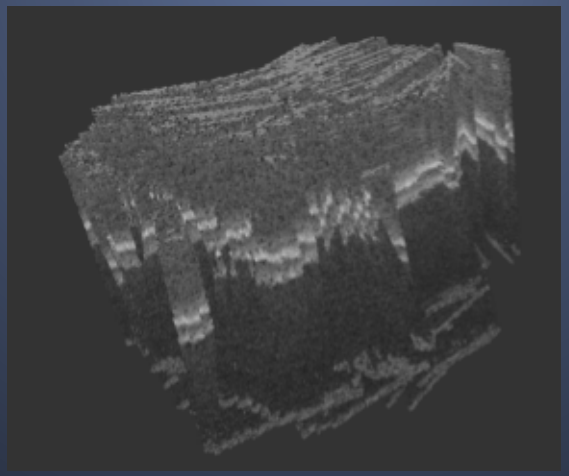


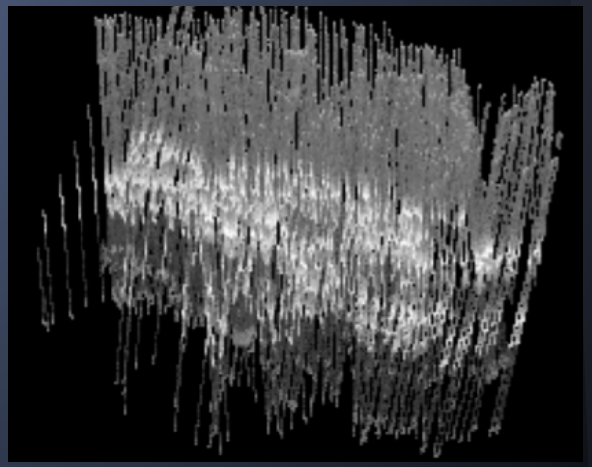
Mais il existe aussi d'autres algorithmes :
Distance Weighted (DW) et la Fast Marching Method (FMM)
Rohling *et al.* **[2]**

4) PNN - Bin-filling

- Tous les pixels sont traversés et affectés au voxel le plus proche
- Moyennage si redondance d'information



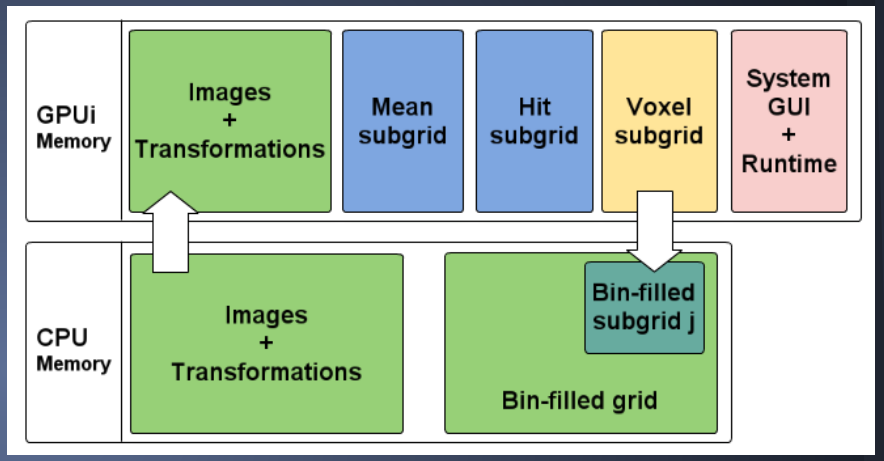




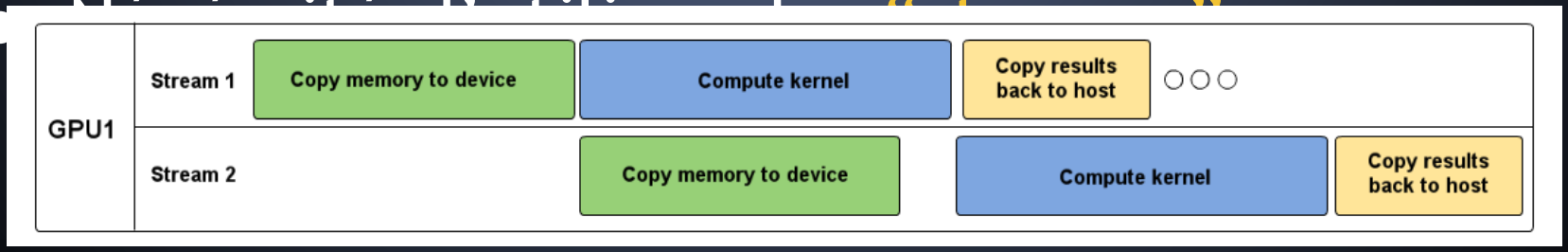
Implémentation sur GPU

Agencement mémoire :

4 grilles de voxels
+ images US
+ transformations

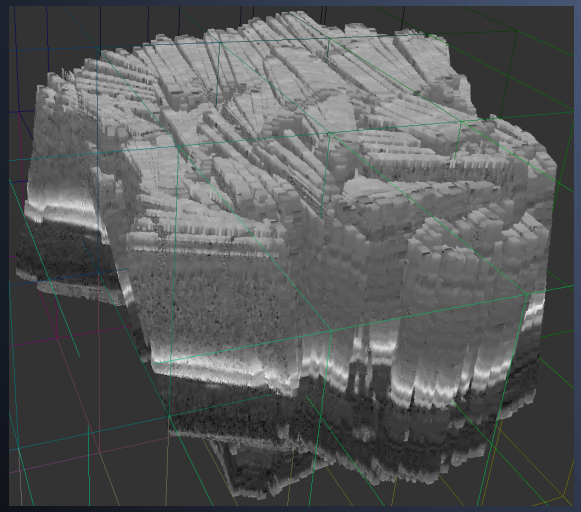


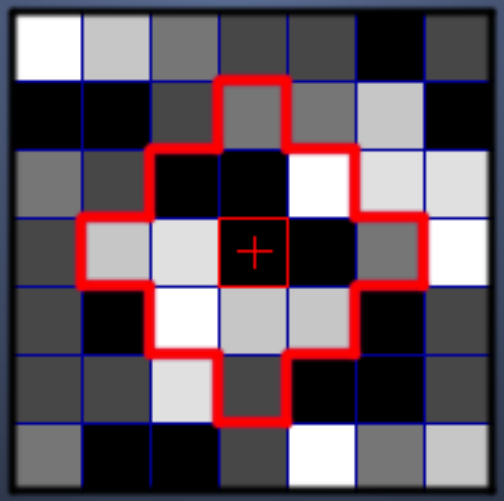
- Transfert de mémoire coûteux (CPU<=>GPU)

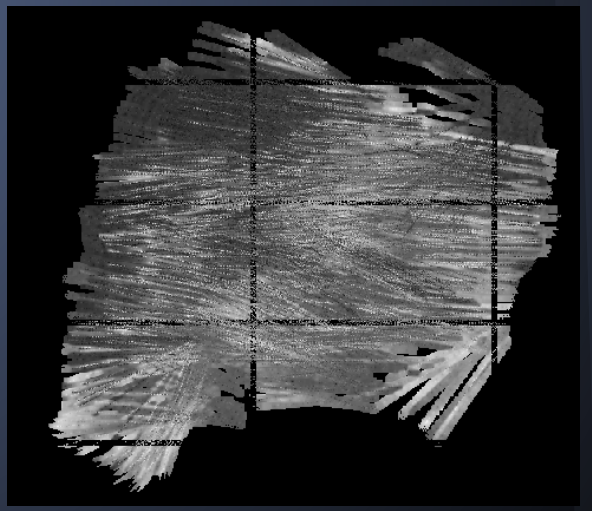


PNN - Hole Filling

- Le bin-filling laisse des trous
- Interpolation des voxels vides avec leurs voisins (région sphérique)

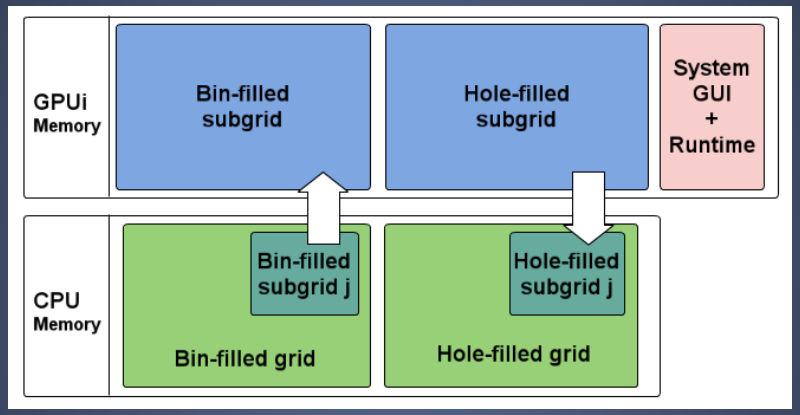






Implémentation GPU

- Agencement mémoire plus simple (deux grilles) :



- Utilisation de la mémoire partagée
- Problème avec les bords des sous-grilles :

    1) Gérer les bords sur le CPU

    2) Prendre des sous-grilles plus grandes sur les GPUs

5) Résultats obtenus

Pour 1610 images (demi plateau tibial)

Taille voxel :	0.5mm	0.2mm	0.1mm
Taille grille :	128x128x256	256x512x512	512x1024x1024
N=4,R=0 :	1.57s	1.62s	2.64s
N=4,R=1 :	1.89s	10.8s	88.4s
N=4,R=2 :	1.96s	12.5s	109.5s
N=4,R=3 :	2.36s	16.5s	146.2s
N=16,R=3 :	6.24s	15.1s	129.9s
N=64,R=3 :	2.98s	14.8s	105.9s
N=256,R=3 :	6.22s	15.4s	87.9s

Bilan et conclusion

- Objectifs atteints
- Scaling efficace
- Contrainte => atout
- Pas assez de temps pour tout implementer
- Pas d'implémentation de référence sur CPU
- Ouvre la voie à d'autres algorithmes plus robustes (FMM)

Références:

[1] T. Wen, Q. Zhu, W. Qin, L. Li, F. Yang, Y. Xie, and J. Gu, "An accurate and effective fmm-based approach for freehand 3d ultrasound reconstruction," Biomedical Signal Processing and Control, vol. 8, no. 6, pp. 645–656, 2013

[2] R. Rohling, A. Geel, and L. Berman, "A comparison of freehand three-dimensional ultrasound reconstruction techniques," Medical Image Analysis, vol. 3, no. 4, pp. 339–359, dec 1999.

Des questions ?