SIMSURF

Vers une simulation réaliste des états de surface par calculs massivement parallèles sur processeurs graphiques

Félix ABECASSIS, Sylvain LAVERNHE,
Pierre-Alain BOUCARD, Christophe TOURNIER

Institut FARMAN 26/11/2013

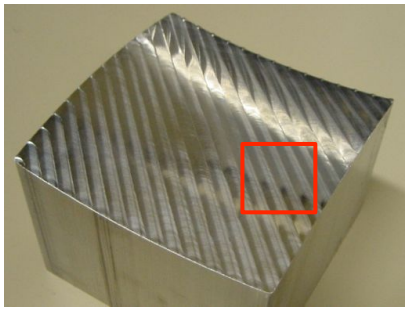
Contexte et objectifs

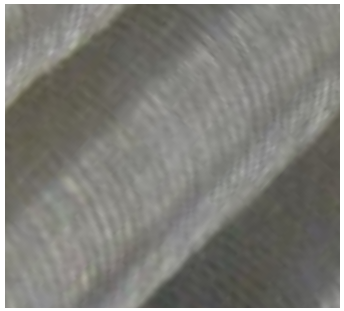
Contexte : Simulation des topographies de surface / fonctions techniques

Moteurs (frottement, étanchéité, intégrité de surface)

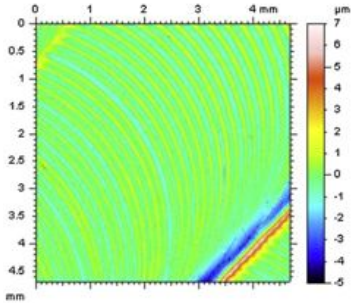
Prothèse médicale (frottement, innocuité)

Verres solaires (optiques)

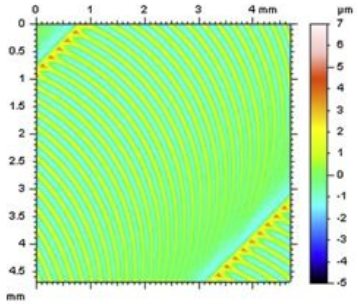




Mesure



Simulation



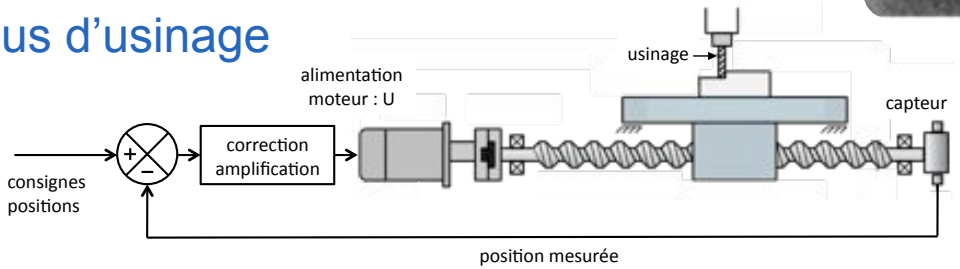
Ecarts géométriques de l'ordre du µm, temps de simulation ordre de 10 minutes

Contexte et objectifs

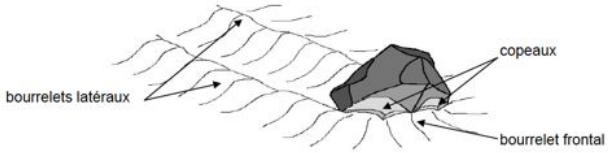
Besoins : Simulation réaliste des topographies de surfaces usinées par plusieurs opérations successives

- Géométrie réelle des outils (dents, usure, bris, etc.)

- Ecarts dus au processus d'usinage
  - numériques,
  - géométriques,
  - mécaniques



- Simulation des surfaces fabriquées par des procédés d'abrasion
  - Chaque grain est un outil, maillage des grains (diamètre 50 µm à 1 µm)



Contexte et objectifs

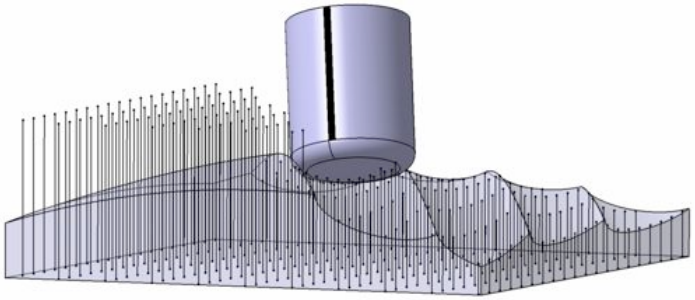
Objectifs

- Minimiser le temps de calcul
  - Organisation et optimisation du code existant pour exploiter au mieux l'architecture massivement parallèle du GPU tout en conservant la précision des résultats
- Réaliser une interface graphique
  - Configurer le calcul, exécuter le calcul et visualiser les résultats dans le même logiciel
- Gestion multi-échelles
  - Zoom sur des portions de pièce calculé en temps réel

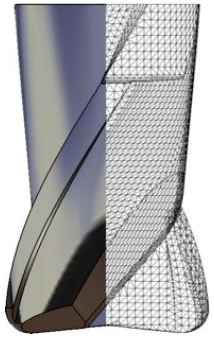
Problématique

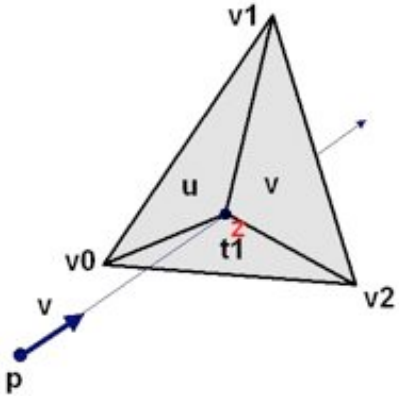
Présentation du calcul

- Partition de l'espace en une grille de N x M droites
  - Outil discrétisé en T triangles
    - Trajectoire avec P positionnements
- Complexité : N x M x T x P
- Exemple : N=M=1024 ; T=10^4 ; P = 10^5
- ➢ Nb opérations ≈ 10^{15}
- ➢ Occupation mémoire > 10 Go



- **Problème adapté au calcul parallèle**
- **Opération élémentaire relativement simple**





Architecture CUDA

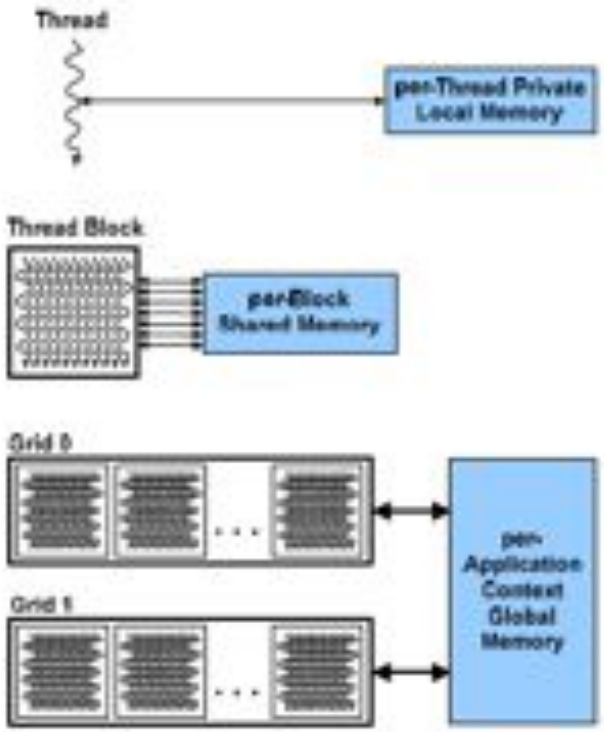
CUDA (Compute Unifed Device Architecture)

Architecture logicielle et matérielle qui permet au cartes graphiques Nvidia d'exécuter des programmes écrits en C, C++ ou autres langages.

- Chaque thread exécute une instance du programme et possède un identifiant
- Chaque thread accède à un espace mémoire réservé réduit mais très rapide

- Les threads peuvent être organisés en blocs et coopèrent entre eux par synchronisation
- Tous les threads d'un même bloc partagent un espace mémoire commun (plus lent)

- Tous les blocs accèdent à la mémoire globale (encore plus lent)



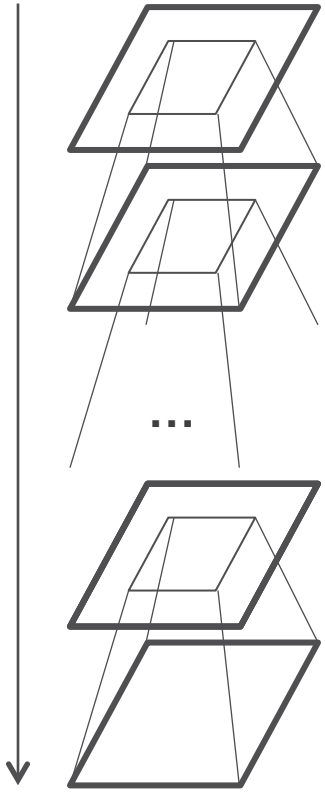
Simulation multi-échelles

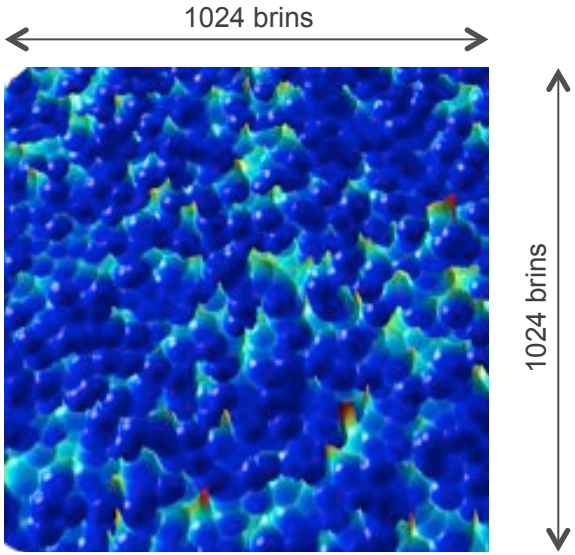
Calculs grande échelle ($P=2.10^5$, $T=2.10^3$) (macro géométrie)

- Thread à granularité élevée
- 1 thread = 1 Triangle * Brins opér.
- Speed-up 64-bit : x5

x64 zoom

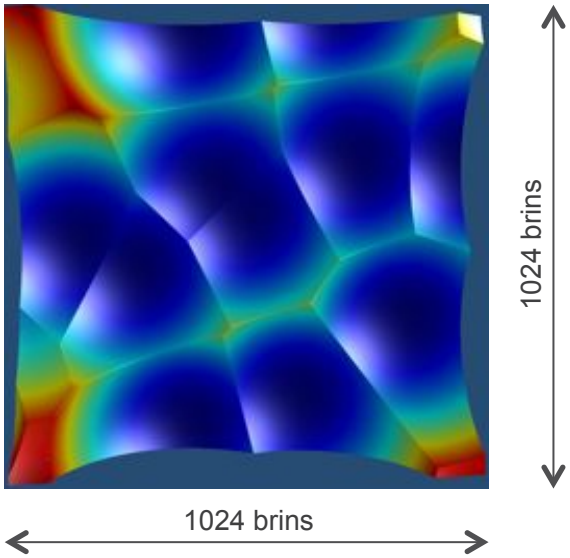
Scroll down = x2
(re computation time : 1000 ms)





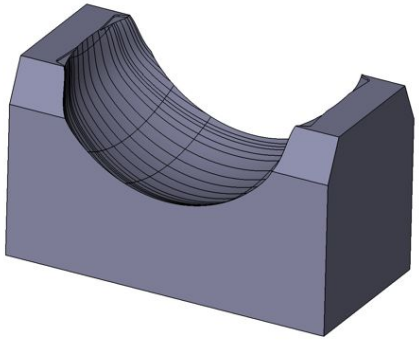
Calculs petite échelle (micro géométrie)

- Thread à granularité élevée
- 1 thread =1 Brin
- Speed-up 64-bit : x4



Application pratique

- **Moule d'injection plastique de verres solaires en polycarbonates**

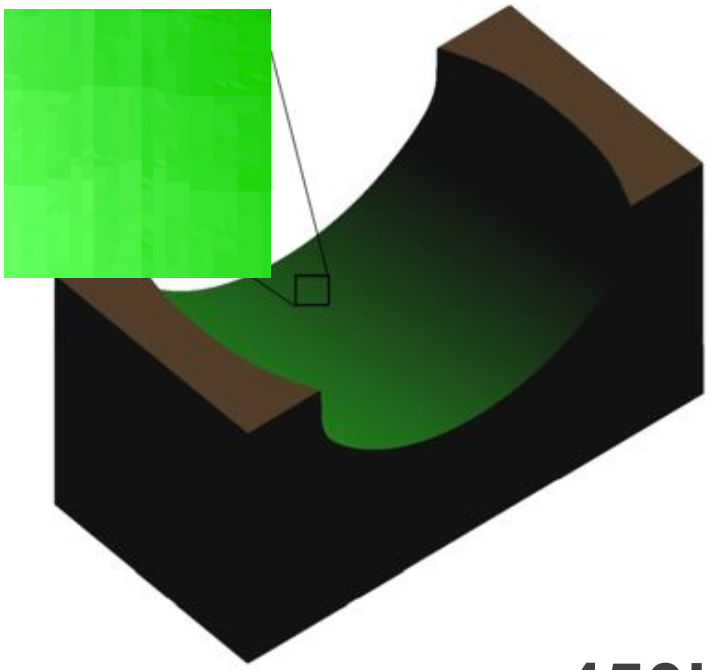


- **Pièce non conforme**
  - **Zone floue au milieu du masque détectée une fois le masque assemblé et testé**

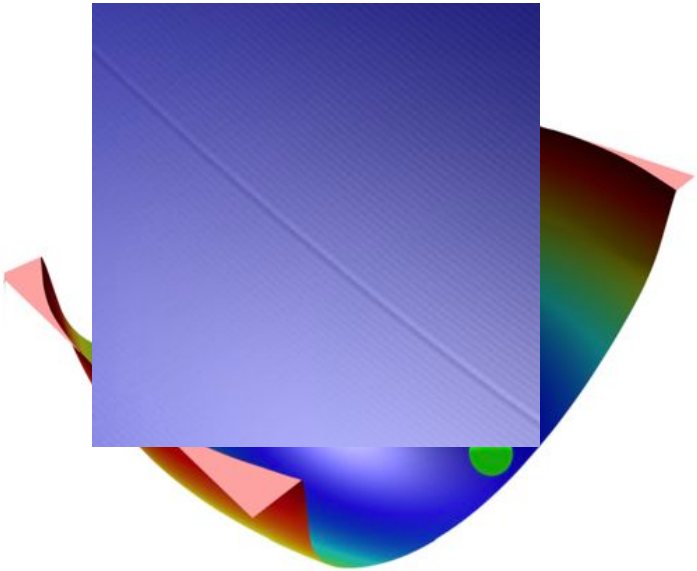
Application pratique

Résultats de simulation *(700 000 positions outils)*

Logiciel Fao (Catia V5)



Zbuffer-GPU simulation



450” **44”**

Calculs simple et double précision

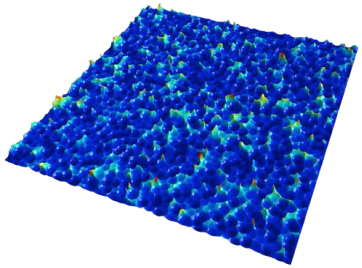
Comparatifs des deux configurations

- **CPU 1 : Intel Xeon X5650 - 2.67GHz**
  - 6 coeurs physiques, 12 coeurs virtuels (hyperthreading)
- **CPU 2 : AMD Phenom II X4 955 - 3.2 GHz**
  - 4 coeurs physiques, 12 coeurs virtuels (hyperthreading)
- **GPU 1 : GeForce GTX 560 Ti – 1.76GHz**
  - 8 multiprocesseurs, 384 cœurs CUDA
- **GPU 2 : NVIDIA Quadro 4000 - 0.95GHz**
  - 8 multiprocesseurs, 256 cœurs CUDA
-

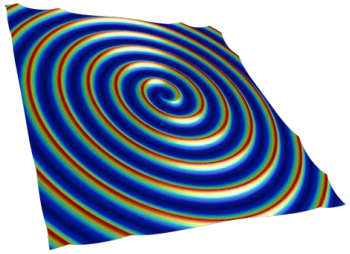
Calculs simple et double précision

Amélioration des performances en simple précision

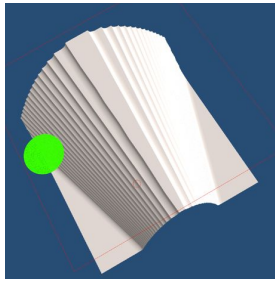
	GeForce GTX 560 Ti		Quadro 4000	
	32-bit	64-bit	32-bit	64-bit
Intel Xeon (6 coeurs)	10,2	2,6	4,4	2,9
AMD Phantom (4 cœurs)	35,5	9,0	15,1	10,0



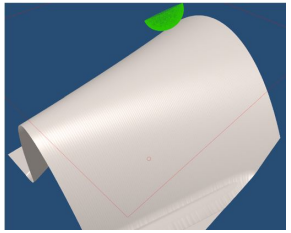
Random



Spiral



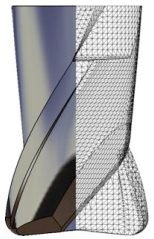
Ebauche



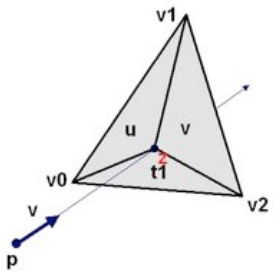
Finition

Calculs simple et double précision

En simple précision, effectuer un test d'intersection entre un rayon et un triangle éloigné peut générer une erreur numérique importante.



1. Chaque triangle du maillage outil est amené à sa position réelle le long de la trajectoire : rotation propre + position/orientation
2. Calcul des coordonnées barycentriques pour vérifier si le point d'intersection avec le rayon est bien à l'intérieur du triangle.



Soit le triangle **t**1 constitué des 3 sommets :

a = (0, 0, 2), **b** = (0, 3.1, 5.1), **c** = (3.1, 0, 7)

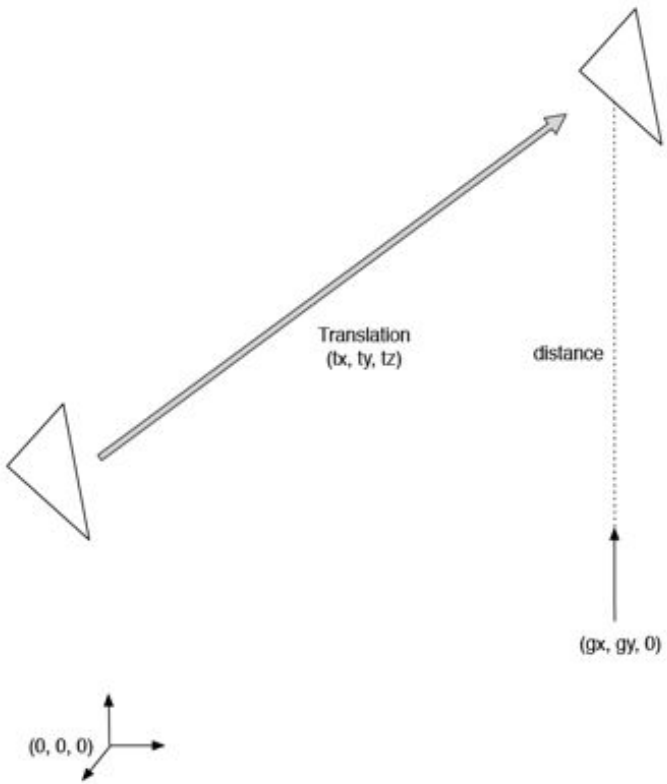
Soit **t**2 le même triangle après une translation de vecteur (**d, d, d**), **d** = 1024

Brin **r**1 = (1.4, 1.4, 0) et brin **r**2 = **r**1 + (**d, d, d**)

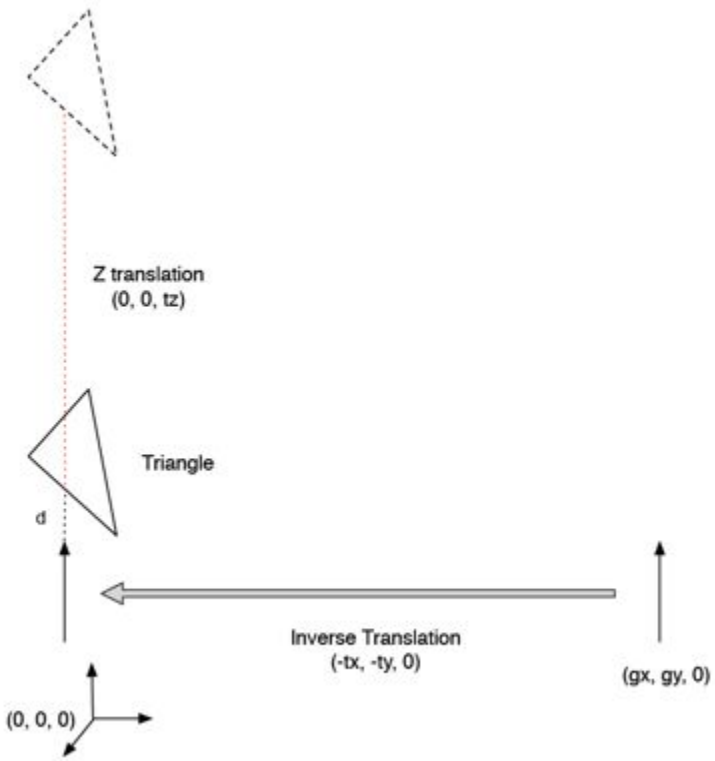
	$e_1.y$	det	u	v	dist
t_1/r_1	3.09999990	9.60999965	3.41000008	4.33999967	5.35806465
t_2/r_2	3.09997558	9.60984897	3.40989756	4.34004163	$d + 5.35803223$

Calculs simple et double précision

Perte de précision



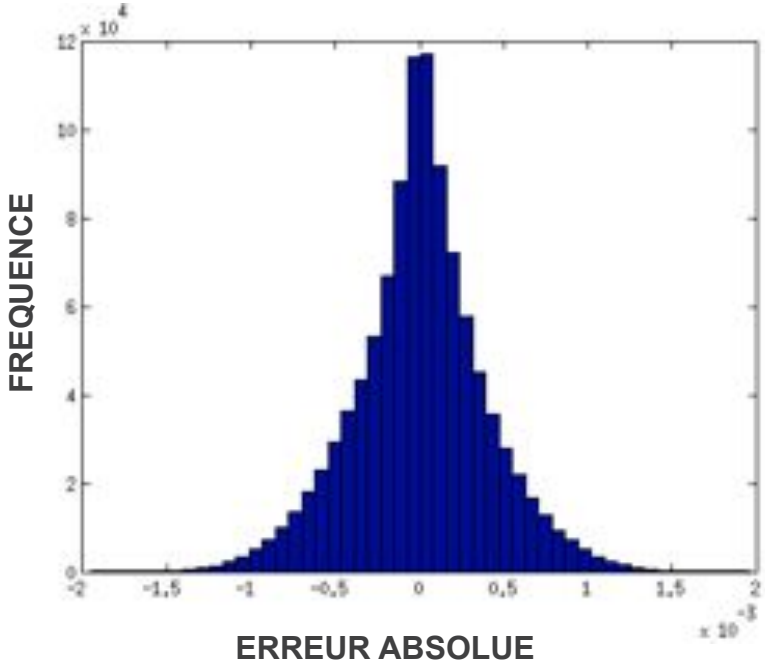
Version initiale



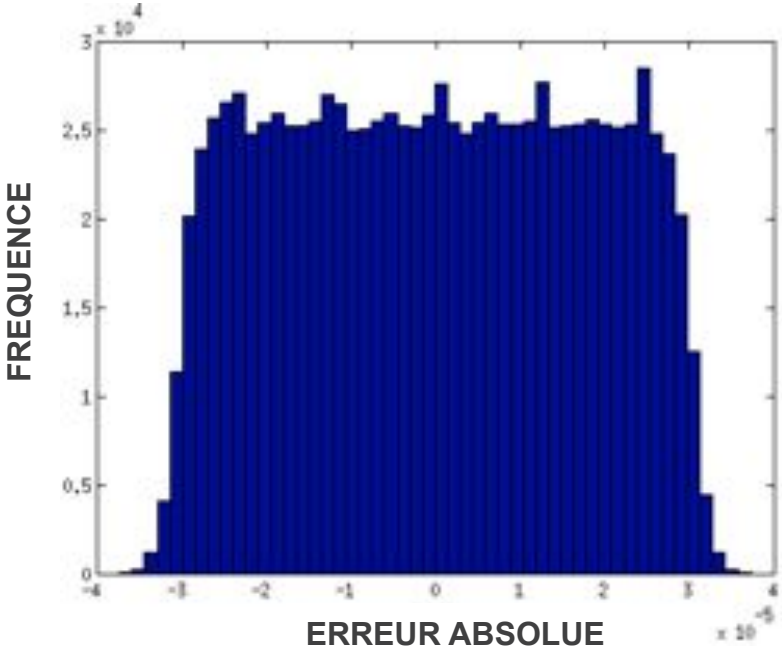
Version améliorée

Calculs simple et double précision

Version initiale



Version améliorée



Erreur absolue sur le Zbuffer par rapport à l'implémentation 64-bit

Conclusions

- **Prototypage d'un logiciel beaucoup plus rapide sur du matériel standard** *(speed-up x100 sur une implémentation mono coeur)*
  - Compétences pointues en informatique pour tirer les meilleures performances des GPU
  - Les optimisations ont également profité à la version CPU multi cores
  - Interface graphique environnement Linux
  - Distribution automatique des calculs sur les threads selon la taille du problème et le hardware.
  - Scrutation en simple précision et rendu final en double précision

1 stage de Master 2,

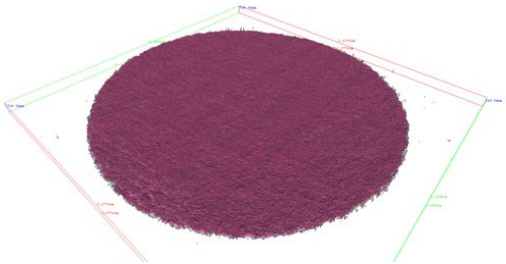
1 conf. internationale (Ingegraf 2013),

1 article « Têtes chercheuses »,

1 article en revue internationale en cours de rédaction (CAD)

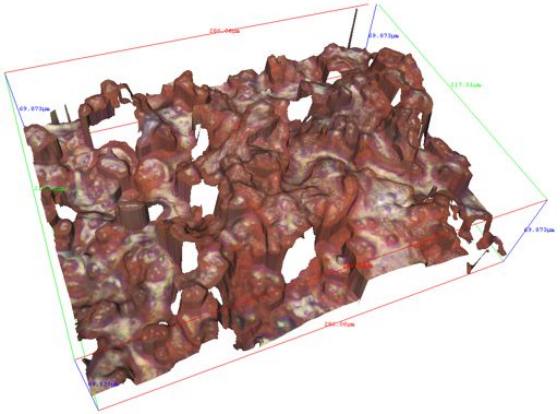
Perspectives

- **Passer à l'échelle inférieure pour simuler les états de surface produits par abrasion**
  - Difficulté de mesure pour l'identification de la géométrie outil par moyens optiques
  - Réparation du maillage nécessaire pour une représentation réaliste sans artefact
  - Taille du maillage très importante
- **Implémenter un modèle stochastique pour simuler des pertes de particules abrasives, intégrer leur usure**



Objectif X5 :
Résolution verticale : 101 nm
Résolution latérale : 14 µm

Objectif X20 :
Résolution verticale : 48 nm
Résolution latérale : 2,9 µm



Objectif X50 :
Résolution verticale : 21 nm
Résolution latérale : 2,13 µm

Perspectives

- **Temps de simulation trop long car taille du problème trop importante : discrétisation trajectoire** x **modèle outil**
  - Utiliser des clusters de cartes ou des cartes plus performantes
    - Quatre cartes sur le cluster du LMT ou n GeForce GTX
  - Modifier le moteur de calcul pour utiliser des librairies natives NVIDIA® OPTIX™ RAY TRACING ENGINE

➢ **SIMSURF 2**