Recette

Dynamic Sampling and Rendering of Algebraic Point Set Surfaces

William Caisson Xavier Chalut Christophe Claustre Thibault Lejemble

Client : Nicolas Mellado

CONTEXTE

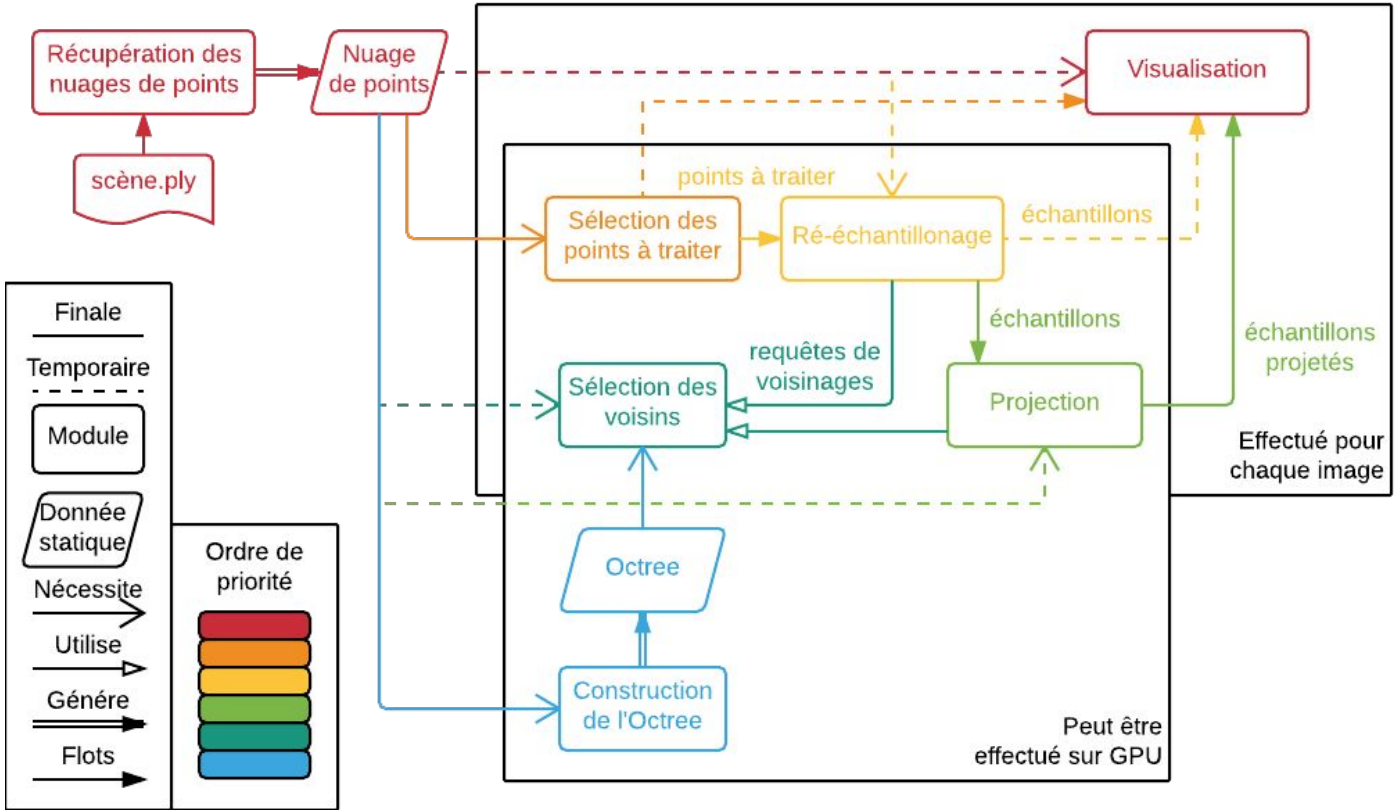
Objectif

Visualiser en temps réel une surface lisse qui approche le nuage de points

Dynamic Sampling and Rendering of Algebraic Point Set Surfaces

Gaël Guennebaud Marcel Germann Markus Gross

Rappel du système



SOMMAIRE

Présentation du programme

Description du plugin

Couverture des exigences

Tests fonctionnels

Déroulement du chef d’oeuvre

Planning et développement

Difficultés rencontrées

Resultats

Démonstration

Visuels et performances

Propositions d'améliorations

Présentation du programme - Description du plugin

Ajout dans Radium :

- Accessibilité depuis un plugin du viewer (+ caméra, renderers ...)
- Ajout d'un renderer possible via le viewer

nécessaire pour qu'un plugin puissent ajouter un renderer

- Support des points dans Radium (classe Mesh)
- Ajout de la propriété "Splat's Radius"
- Ajout des "Pointy" RenderObjects

supports des points & visualisations

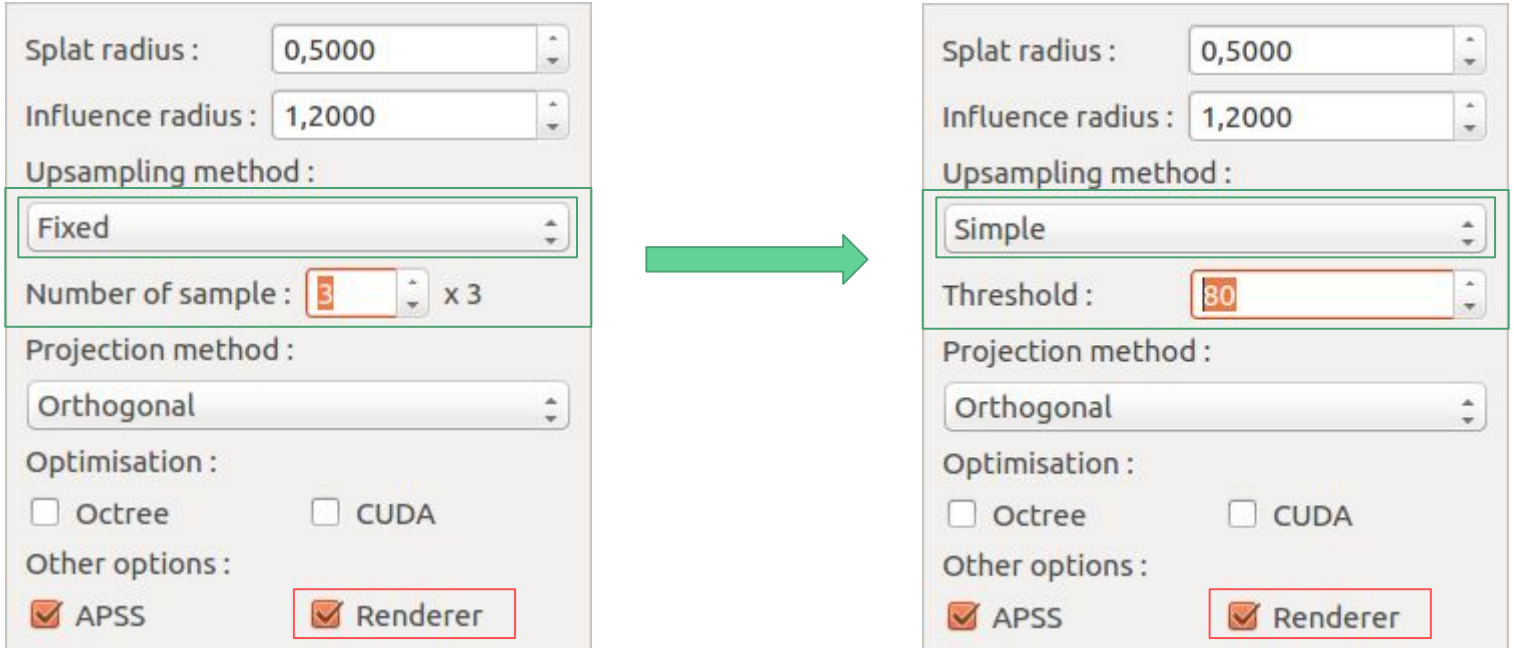
- Shaders pour le rendu de point
  - 1 couple vertex/fragment shader
  - 1 triplé vertex/geometry/fragment shader

Présentation du programme - Description du plugin

Fonctionnalités du plugin :

- Renderer basique de points
- Instanciation de “Pointy” RenderObjects automatique à l’ouverture d’un fichier
    - ➢ lecture des fichiers déjà présente dans Radium
- Implémentation de l’article incomplète, cependant personnalisable :
    - différentes méthodes de rééchantillonnage
    - différentes méthodes de sélection des voisins
    - implémentation CUDA optionnelles de certaine étapes
    - parallélisation de certaine étapes sur CPU (OpenMP)

Présentation du programme - Description du plugin



Présentation du programme - Couverture des exigences

P1	Le travail effectué doit s'intégrer dans le moteur de rendu Radium-Engine sous la forme d'un plugin	
P2	Le plugin doit pouvoir fonctionner sur un système d'exploitation Linux qui comporte une carte graphique NVIDIA permettant l'exécution de programmes CUDA	
P3	Le plugin doit permettre à Radium-Engine d'afficher un nuage de points	
P4	Le plugin doit permettre à Radium-Engine d'extraire un nuage de points des fichiers .PLY donnés en entrée	
P5	Le plugin doit permettre à Radium-Engine d'afficher une scène décrite en nuage de points de moins de 1 million de points avec une fréquence d'affichage minimum de 50 images par seconde	
P6	Il devra être fourni lors de la recette, le code source du plugin	
P7	Il devra être fourni lors de la recette, une documentation utilisateur complète décrivant comment utiliser le plugin	

Présentation du programme - Couverture des exigences

S1	Le plugin doit permettre à Radium-Engine d'afficher une scène décrite en nuage de points de moins de 2 millions de points avec une fréquence d'affichage minimum de 50 images par seconde	
S2	Quelque soit le positionnement et l'orientation de la caméra, la surface des objets dans le champ de vision doivent présenter aucune ouverture de plus que celle due à la description du nuage de points	
S3	Le plugin doit permettre à Radium-Engine d'afficher une surface lisse à partir d'un nuage de points	
S4	Le plugin ne doit pas empêcher Radium-Engine de fonctionner sous Windows et MacOS	

Présentation du programme - Tests fonctionnels

Nom du scénario	Actions à effectuer	Résultats attendus	Exigences
Test plugin	Suppression de notre plugin de Radium-Engine	Le moteur doit fonctionner comme à l'initial	P1
Test lecture d'un fichier .PLY	Exécuter Radium-Engine, charger le fichier .PLY, enregistrer le nuage de points chargé dans un fichier .PLY	Le contenu du fichier .PLY issu de la sauvegarde doit être le même que le contenu du fichier .PLY qui a servi au chargement	P4
Test affichage d'un nuage de points	Exécuter Radium-Engine , charger le fichier .PLY	L'affichage du nuage de points doit se faire correctement par rapport aux données du fichier .PLY	P3
Test taux de rafraîchissement	Exécuter Radium-Engine , charger le fichier .PLY de 1 million de points	Le nombre d'image par seconde indiqué doit être au minimum de 50.	P5
Test taux de rafraîchissement 2	Exécuter Radium-Engine , charger le fichier .PLY de 2 million de points	Le nombre d'image par seconde indiqué doit être au minimum de 50.	S1
Test non création de trous dans les surfaces	Exécuter Radium-Engine , charger les fichiers .PLY un à un	Les surfaces des objets doivent présenter aucune ouverture de plus que celle dues à la description des nuages de points	S2
Test continuité des surfaces	Exécuter Radium-Engine , charger le fichier .PLY	La surface des objets affichés doit être visuellement lisse	S3

Déroulement du chef d'oeuvre - Planning et développement

Outils:

Organisation:

- Travail en groupe par tâche
- 5 réunions client

Temps de travail : 230h/personne

Déroulement du chef d’oeuvre - Planning et développement

Développement

Deux branches Git

Branche Master :

- 220 commits
- ~2500 lignes de codes pour le plugin PointyCloud
- 43 nouveaux fichiers
- 20 nouvelles classes

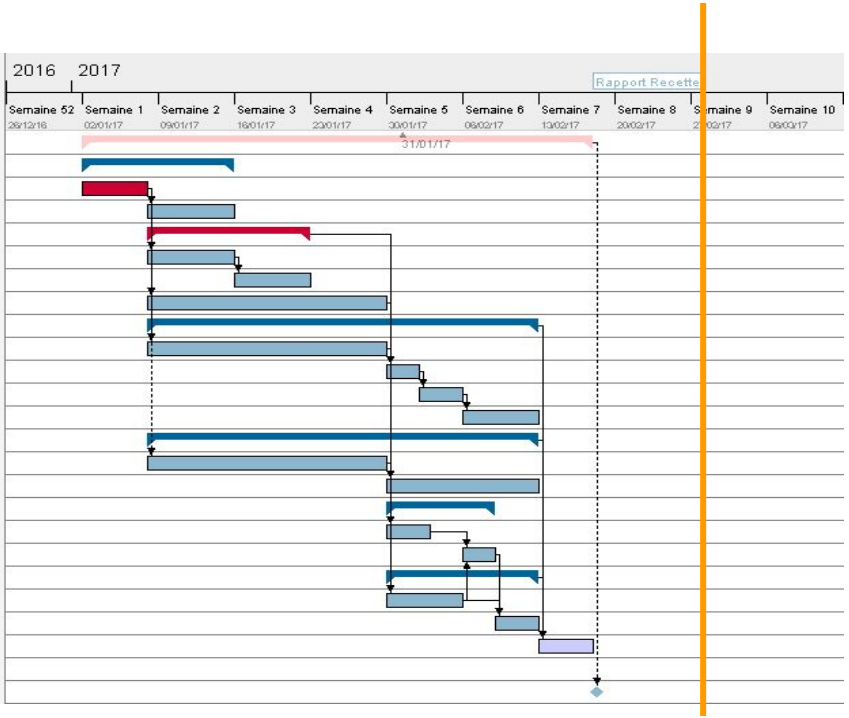
Branche CUDA :

- 49 commits
- ~630 lignes de codes
- 13 nouveaux fichiers

Déroulement du chef d'oeuvre - Planning et développement

Planning prévisionnel

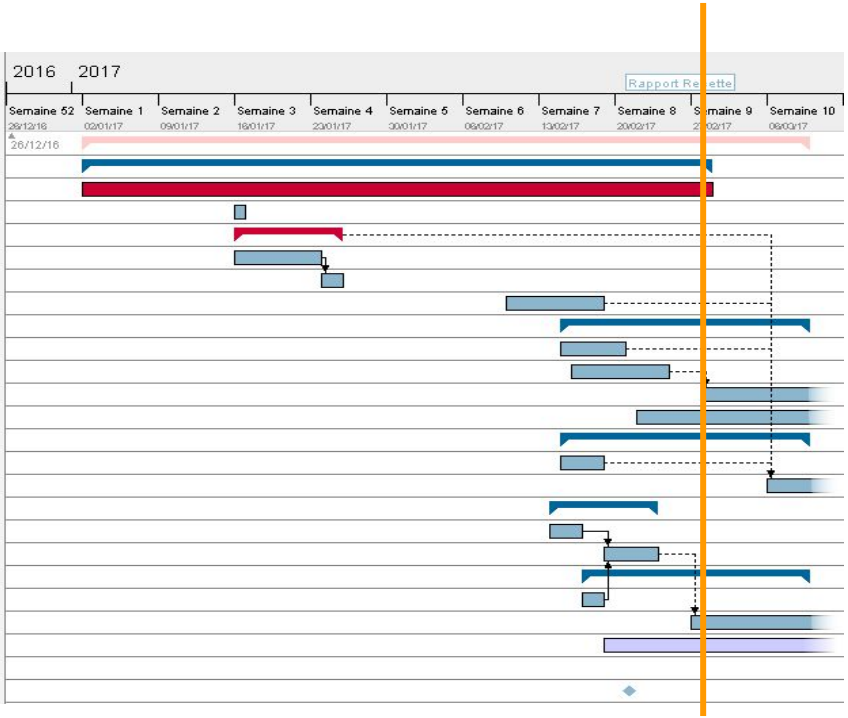
Nom	Date de début	Date de fin
Développement	02/01/17	17/02/17
Récupération des nuages de point	02/01/17	15/01/17
Représentation des nuages de points dans Radium	02/01/17	07/01/17
Lecture des fichiers .PLY en entrée de Radium	08/01/17	15/01/17
Visualisation	08/01/17	22/01/17
Visualisation des points	08/01/17	15/01/17
Gestion de l'occultation	16/01/17	22/01/17
Sélection des points à traiter	08/01/17	29/01/17
Ré-échantillionnage	08/01/17	12/02/17
Génération d'un nombre fixé d'échantillons	08/01/17	29/01/17
Calcul simple du nombre d'echantillon	30/01/17	01/02/17
Calcul complexe du nombre d'echantillon	02/02/17	05/02/17
Génération des échantillons sur GPU	06/02/17	12/02/17
Projection	08/01/17	12/02/17
Projection orthogonale en utilisant la bibliothèque	08/01/17	29/01/17
Projection presque othogonale	30/01/17	12/02/17
Sélection des voisins	30/01/17	08/02/17
Sélection simple	30/01/17	02/02/17
Utilisation de l'Octree	06/02/17	08/02/17
Construction de l'Octree	30/01/17	12/02/17
Par CPU	30/01/17	05/02/17
Par GPU	09/02/17	12/02/17
Tests d'intégration	13/02/17	17/02/17
Reunions client	06/12/16	06/12/16
Rapport Recette	18/02/17	18/02/17



Déroulement du chef d'oeuvre - Planning et développement

Planning effectif

Nom	Date de début	Date de fin
Développement	02/01/17	09/03/17
Récupération des nuages de point	02/01/17	28/02/17
Représentation des nuages de points dans Radium	02/01/17	28/02/17
Lecture des fichiers .PLY en entrée de Radium	16/01/17	16/01/17
Visualisation	16/01/17	25/01/17
Visualisation des points	16/01/17	23/01/17
Gestion de l'occultation	24/01/17	25/01/17
Sélection des points à traiter	10/02/17	18/02/17
Ré-échantillionnage	15/02/17	09/03/17
Génération d'un nombre fixé d'échantillons	15/02/17	20/02/17
Calcul simple du nombre d'echantillon	16/02/17	24/02/17
Calcul complexe du nombre d'echantillon	28/02/17	09/03/17
Génération des échantillons sur GPU	22/02/17	09/03/17
Projection	15/02/17	09/03/17
Projection orthogonale en utilisant la bibliothèque	15/02/17	18/02/17
Projection presque othogonale	06/03/17	09/03/17
Sélection des voisins	14/02/17	23/02/17
Sélection simple	14/02/17	16/02/17
Utilisation de l'Octree	19/02/17	23/02/17
Construction de l'Octree	17/02/17	09/03/17
Par CPU	17/02/17	18/02/17
Par GPU	27/02/17	09/03/17
Tests d'intégration	19/02/17	09/03/17
Reunions client	06/12/16	06/12/16
Rapport Recette	21/02/17	21/02/17



Déroulement du chef d’oeuvre - Difficultés rencontrées

Contraintes matériel:

- Support CUDA

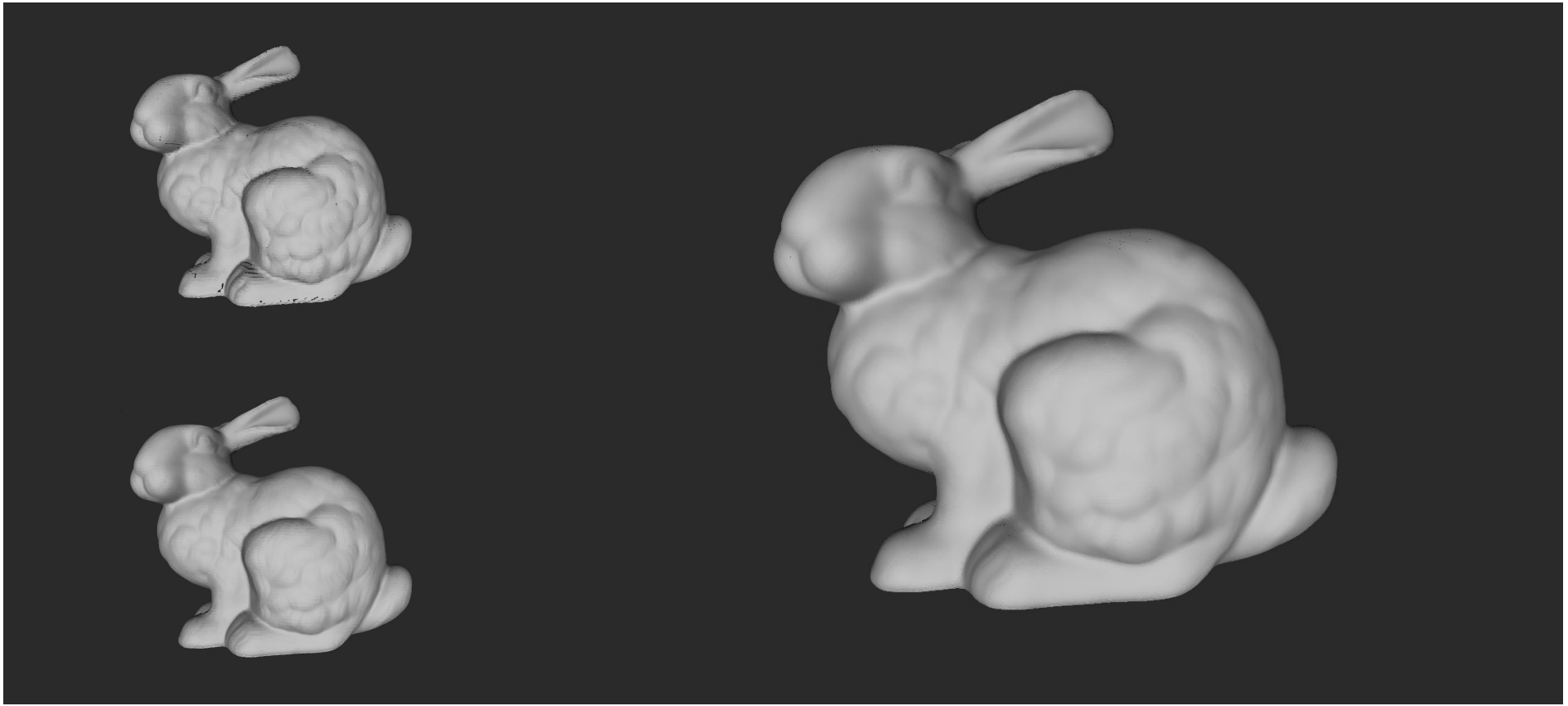
Planning

- Difficultés à se mettre à 100% sur le projet

Programmation

- Retour en arrière sur la conception (structures de données)
- Problèmes de performance

Résultats - Démonstration



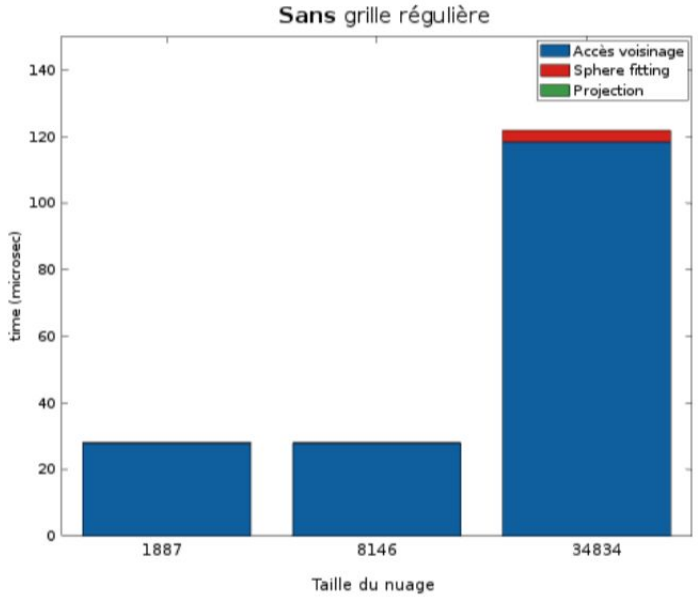
Résultats - Visuels et performances

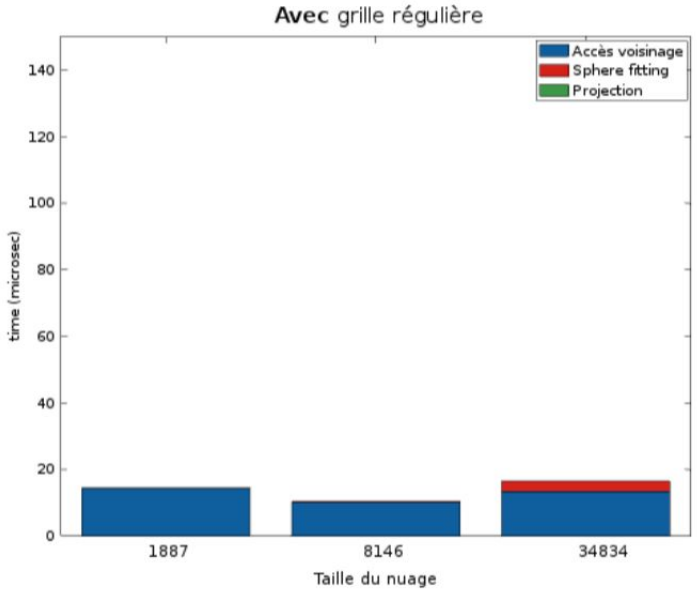
Proportion de temps de calcul entre les différentes étapes du calcul de l'APSS

Culling	2,01%
Upsampling	4,88%
Projecting	90,86%
- Neighbors query	82,93%
- Sphere fitting	17,05%
- Point projection	0,02%
Loading in mesh	2,24%

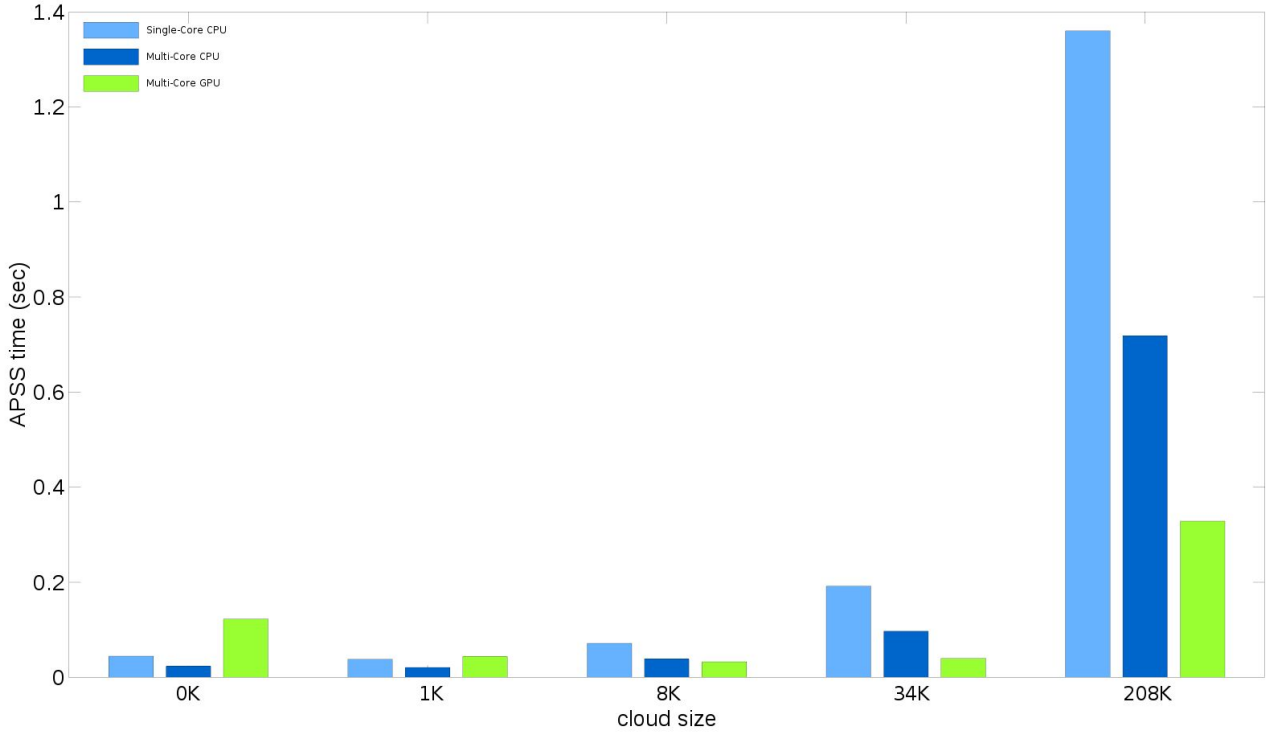
Résultats - Visuels et performances

Utilisation d'une grille régulière : temps d'accès aux voisins

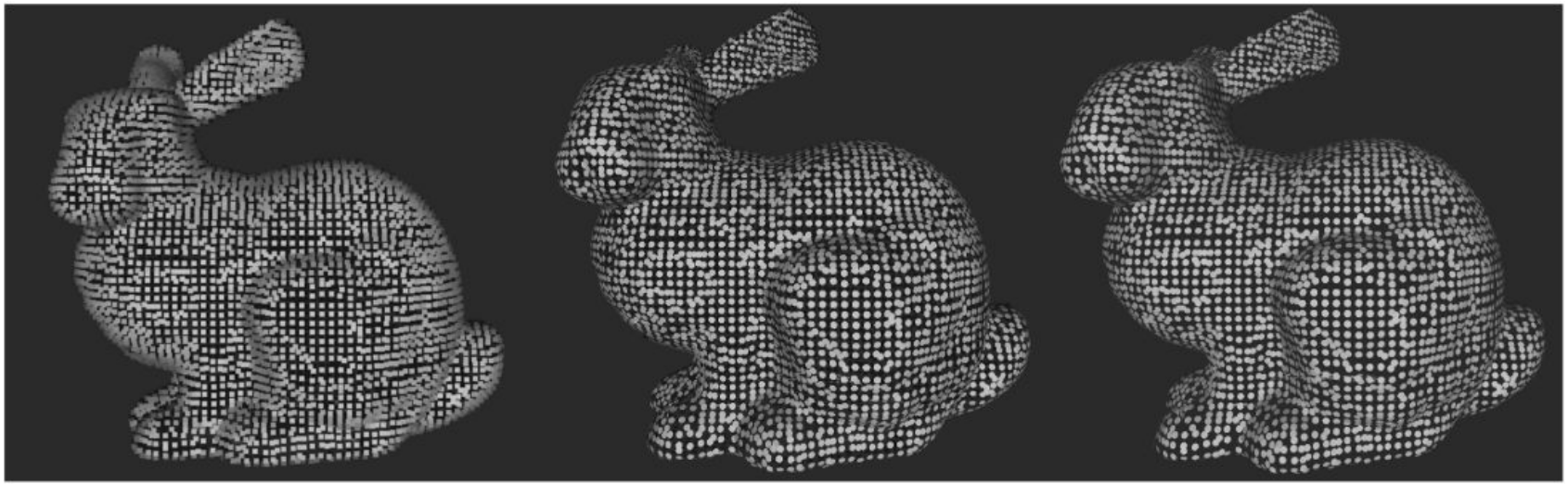




Résultats - Visuels et performances



Résultats - Visuels et performances

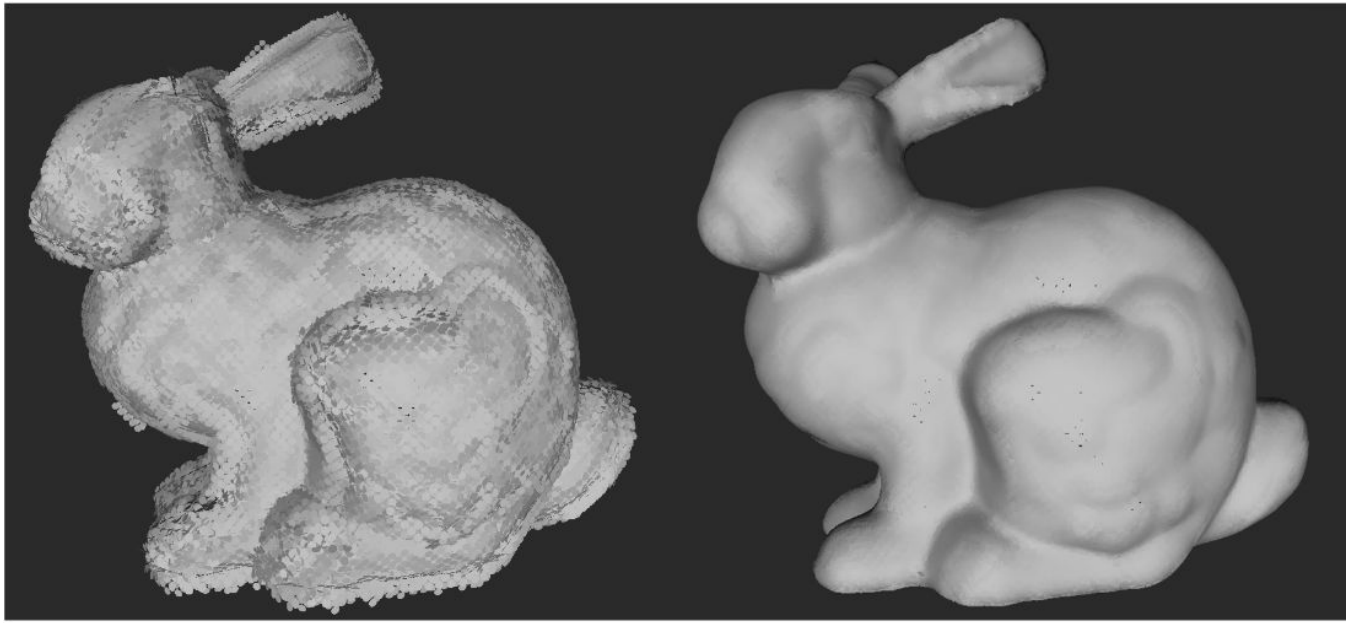


Nuage de points de référence (splats carrées non-orientées)

Splats elliptiques orientées

Culling

Résultats - Visuels et performances



Sur-échantillonnage (fixe)

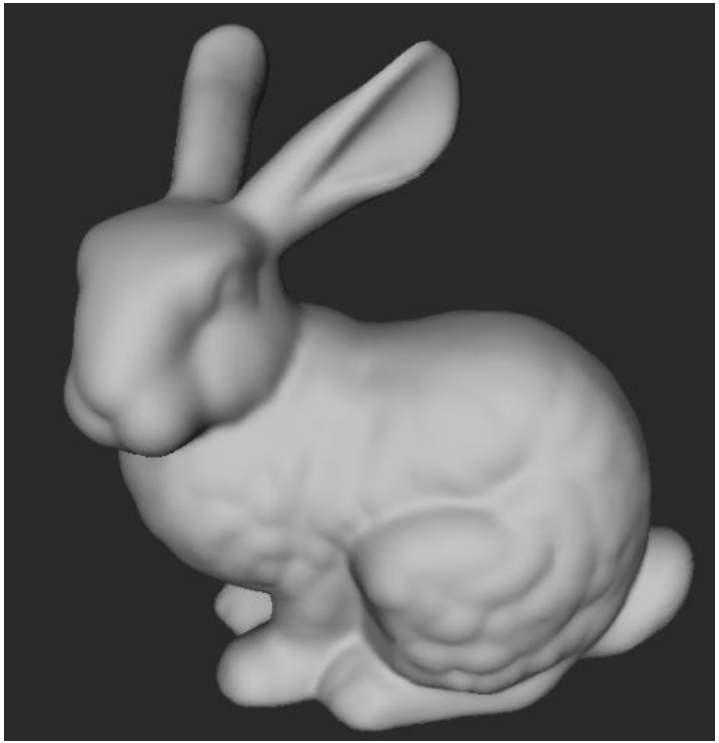
Projection (orthogonale) sur l'APSS

Résultats - Visuels et performances

Lapin : 200k points

Temps réel : Non, ≃ 0,5 frame/sec

Utilisation du multithreading CPU



Résultats - Propositions d’améliorations

- Implémenter les dernières versions de :
  - projection
  - ré-échantillonnage
- Finaliser l’implémentation sur GPU
- Pouvoir gérer plusieurs nuages de points indépendamment
- Sauvegarder un nuage de point après ré-échantillonnage

Merci de votre attention

Avez-vous des questions ?