DE LA RECHERCHE À L'INDUSTRIE

Étude et évaluation de la structure de donnée SVDAG et ses variantes pour le RayTracing en visualisation

Antoine Roche – M1 CHPS
Tuteur CEA : Jérôme Dubois
Tuteur enseignant : Michael Krajecki

08/04/19 – 30/08/19

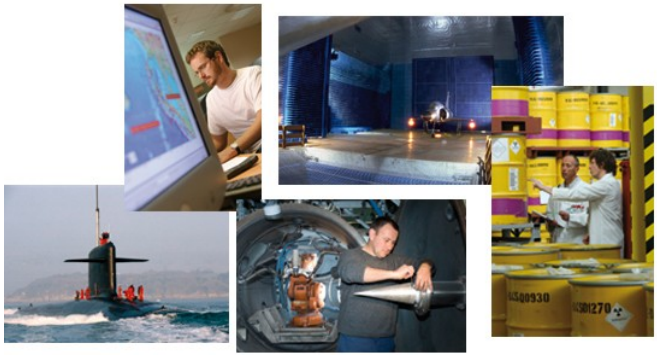
Commissariat à l'énergie atomique et aux énergies alternatives - www.cea.fr
CEA, DAM, DIF, F-91297, Arpajon, France

SOMMAIRE

- Présentation du CEA
- Contexte du stage
- Études menées
- Travail réalisé
- Expérience personnelle et professionnelle
- Bibliographie

Acteur majeur de la recherche, du développement et de l'innovation

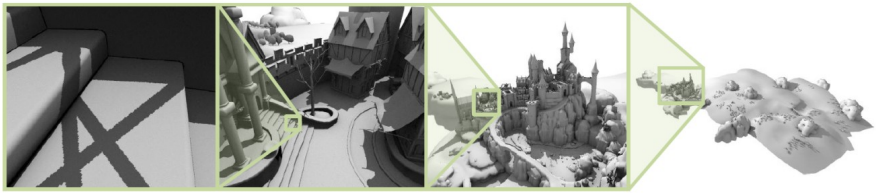
- 4 domaines :
- Défense et sécurité
- Energies bas carbone
- Recherche technologiques pour l'industrie
- Recherche fondamentale

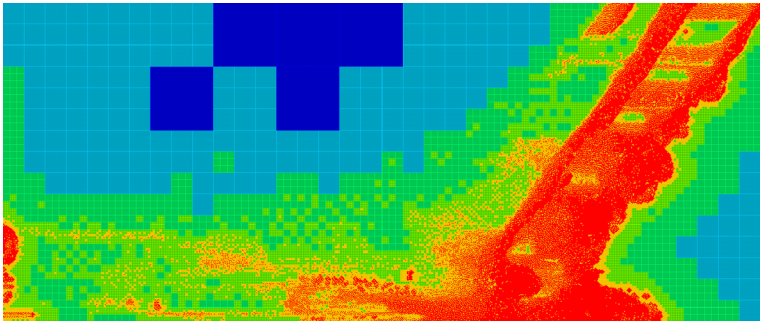


Equipe de visualisation de simulation massive HPC
Bruyère le Châtel (IDF)

- Appliquer des technologies du jeu vidéo sur de la visualisation scientifique

SVDAG* : Compression de scènes voxélisées





VTK-HTG** : Réduction du volume et du temps de rendu en affinant de manière locale

*Sparse Voxel Directed Acyclic Graph

**VTK HyperTreeGrid, tree based Adaptive Mesh Refinement

Principales étapes :

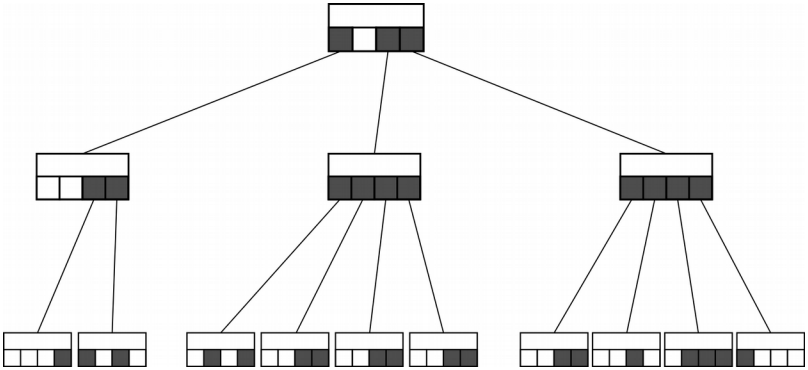
- Etude du SVO* / SVDAG / SSVDAG**
- Maîtrise de la construction HTG
- Convertisseur HTG vers SVDAG / SSVDAG
- Application de rendus SVDAG sur des données HTG

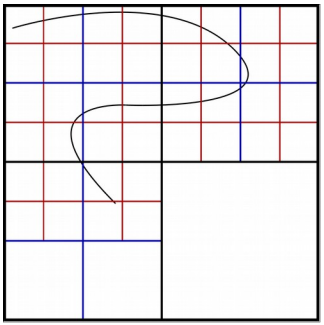
*Sparse Voxel Octree
**Symmetric-aware SVDAG

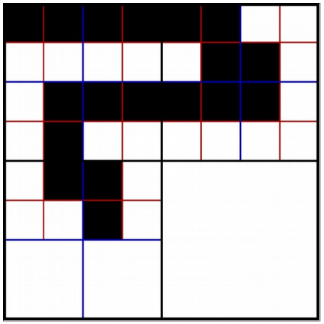
Thèse de Viktor Kämpe : SVDAGs[2]
Nvidia research : ESVO[5]
SVO builder github[6]

Arbre de voxels :

- De niveau défini
- Voxels au dernier niveau
- Actuellement construit sur CPU



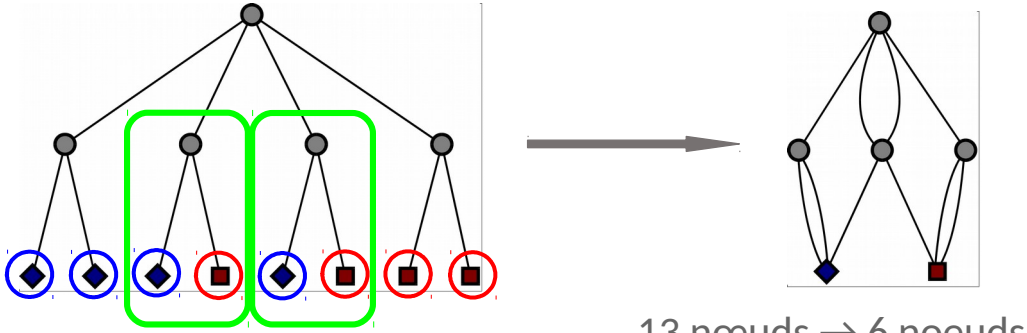




Thèse de Viktor Kämpe : SVDAGs[2]
Viktor Kämpe, Erik Sintorn, and Ulf Assarsson : High resolution sparse voxel dags[3]

Développé par l'équipe de visualisation de l'université de Chalmers (Suède)

Construction et raytracer avec CUDA
Octree compressé par niveau
Plusieurs parents pour un nœud possible
Suppression des nœuds identiques



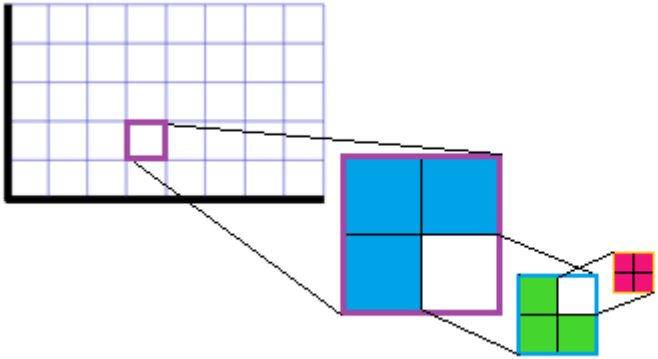
13 nœuds → 6 noeuds

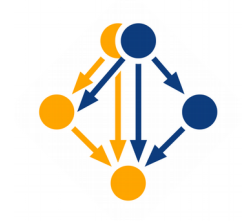
Massimiliano Guarrasi : An introduction to adaptive mesh refinement (AMR)[1]

Grille d'arbres de voxels

Niveau des voxels variable

Affinage adaptatif





```
class Svdag(CMakePackage):
    homepage = "svdag"
    url      = "svdag"

    version('develop', git
example-master")

    depends_on('cmake@2.6.

    depends_on( 'glm' )
    depends_on( 'cereal' )
    depends_on( 'glew' )
    depends_on( 'cuda' )
    depends_on( 'sdl2' )
```

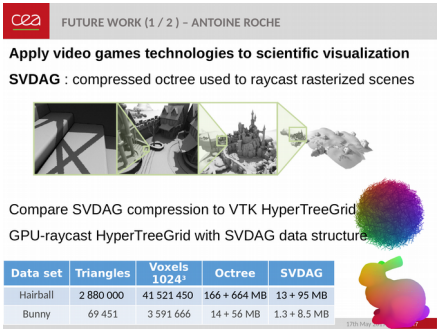
Installation de Spack : gestionnaire de paquets

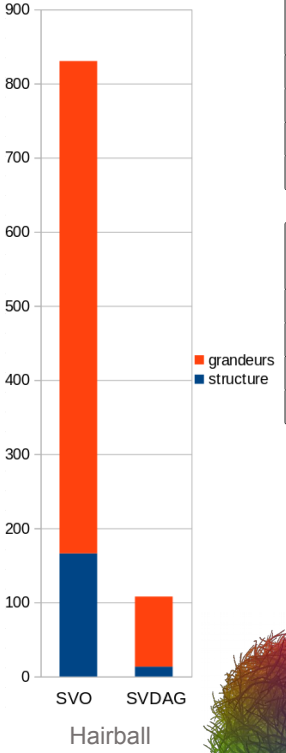
Installation du programme SVDAG avec résolution de compatibilité Windows vers Unix

Création recette Spack du SVDAG

Analyse des performances du SVDAG

Slide présentant mon travail (pour Visu2019 et Eurovis19 notamment)



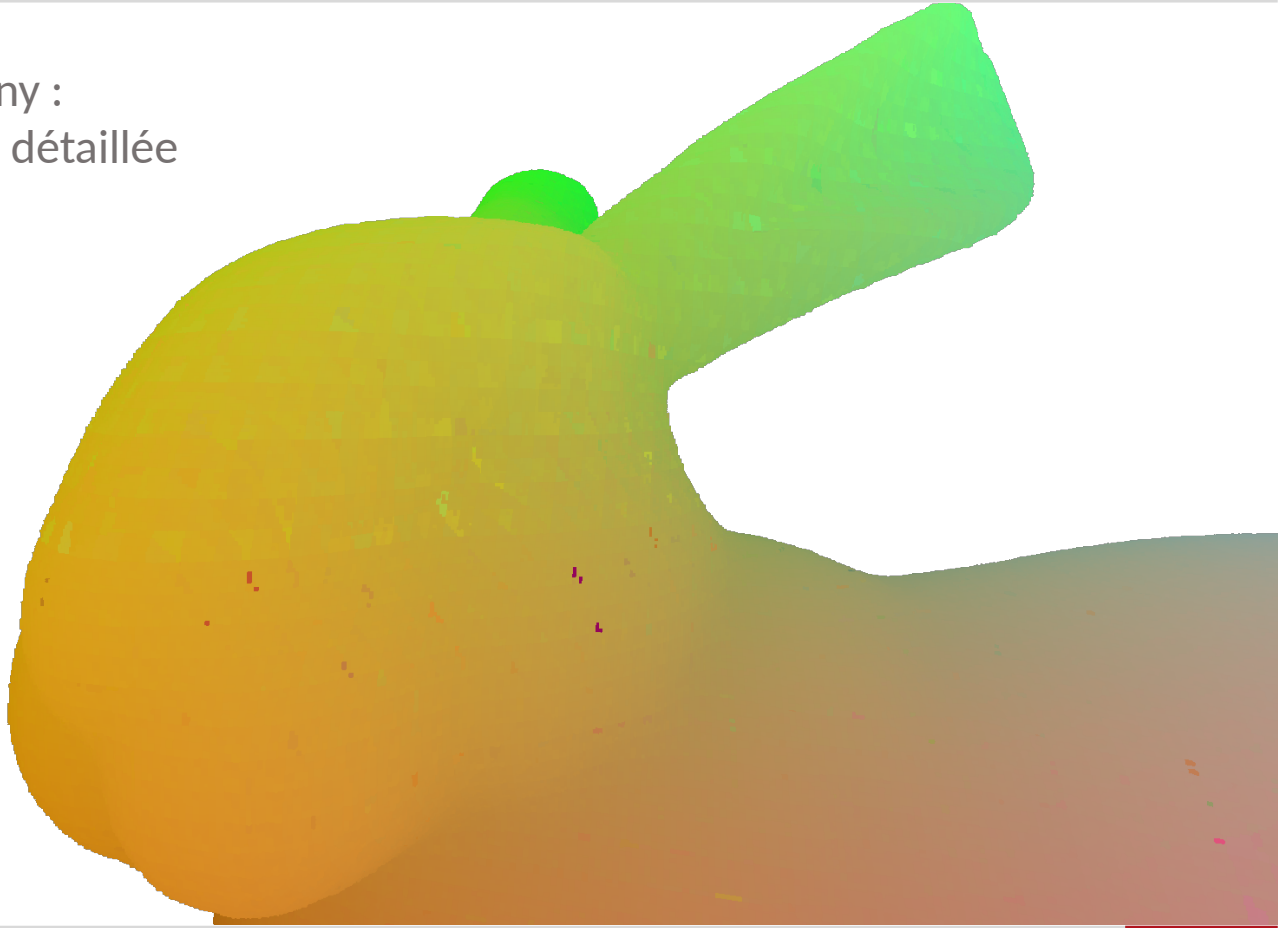


	Nombre Triangles	Taille non-structuré	Temps construction SVO	Temps constrution SVDAG
Bunny	69 451	3.0 Mo	3.9s	4.3s
Dragon	871 414	33.8 Mo	5.7s	3.0s
Hairball	2 880 000	236.1 Mo	58.7s	47.7s
Lucy	28 055 742	533.1 Mo	77.5s	1.7s

	Nombre voxels	Taille SVO Structure + grandeurs	Taille SVDAG Structure + grandeurs	Taux compression
Bunny	3 591 666	14.4 + 57.5 Mo	1.3 + 8.5 Mo	86.3%
Dragon	2 688 970	10.8 + 43.0 Mo	1.0 + 6.4 Mo	86.2%
Hairball	41 521 450	166.1 + 664.3 Mo	13.3 + 94.6 Mo	87.0%
Lucy	1 540 004	6.2 + 24.3 Mo	0.6 + 3.6 Mo	86.3%

10x

Bunny :
Tête détaillée



TRAVAIL RÉALISÉ – IMAGES OBTENUES

Bunny :
Tête détaillée
Lucy :
Détails plus grossiers

Université technique de Chalmers

CEA

Journée Visu 2019

https://journee-visu.github.io/

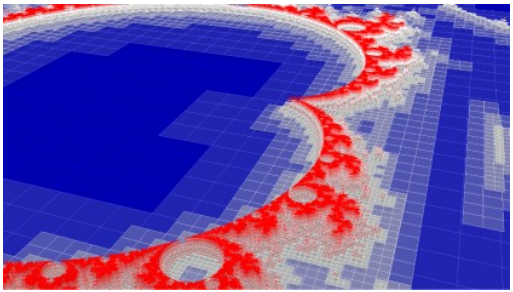
Installations :

- Proxy ⟹ `Error: FetchError: All fetchers failed`
- VTK

SVDAG :

- Compatibilité Windows – Linux
- Pas de documentation

Maitrise du HTG avec VTK
Convertisseur HTG → SVDAG



Fractal de mandelbrot
6 milliards éléments

Tests sur des jeux de données plus riches

CONCLUSION

- Projet demandant beaucoup de recherche
- Domaine nouveau pour moi
- Difficultés liées à un environnement sécurisé
- Résultats encourageants
- Beaucoup d'expérience apportée

BIBLIOGRAPHIE

- [1] Massimiliano Guarrasi. An introduction to adaptive mesh refinement (amr) : Numerical methods and tools, 2015.
- [2] Viktor Kämpe. Sparse Voxel DAGs. PhD thesis, Chalmers University of Technology, 2016.
- [3] Viktor Kämpe, Erik Sintorn, and Ulf Assarsson. High resolution sparse voxel dags. ACM Transactions on Graphics, 32(4), 2013. SIGGRAPH 2013.
- [4] Samuli Laine and Tero Karras. Efficient sparse voxel octrees. In Proceedings of ACM SIGGRAPH 2010 Symposium on Interactive 3D Graphics and Games, 2010.
- [5] Samuli Laine and Tero Karras. Efficient sparse voxel octrees – analysis, extensions, and implementation. NVIDIA Technical Report NVR-2010-001, NVIDIA Corporation, February 2010.
- [6] Jeroen Baert - Out-of-core svo builder https://github.com/Forceflow/ooc_svo_builder
- [7] Dan Dolonius - Dag-example https://github.com/gegoggigog/DAG-example

MERCI POUR VOTRE ECOUTE
DES QUESTIONS ?

Antoine Roche – M1 CHPS

Tuteur CEA : Jérôme Dubois

Tuteur enseignant : Michael Krajecki