Visualisation de calculs CFD : des millions de particules en temps réel avec Snorky3D

Visualisation of CFD computations: millions of particles in real time with Snorky3D

Benoît MATHIEU

English Abstract—This presentation illustrates how to take advantage of the computing and graphics power of a workstation to visualize complex CFD datasets by tracing millions of particles in real time. The presentation describes the optimizations of the trajectory computing engine as well as some features of the rendering engine and some interactivity features. These elements help producing stuningly realistic and intuitive visualizations of CFD datasets.

1 Introduction

Les simulations et visualisations basées sur des particules connaissent un essor rapide depuis que CPUs et GPUs permettent des traitements en temps réel sur des millions de points. La visualisation de résultats de simulations en dynamique des fluides à l'aide de particules est une application connue de ces techniques, dont l'exploitation dans les logiciels de visualisation scientifique est pourtant loin de tirer pleinement parti, que ce soit en termes de performances ou en termes de qualité de rendu.

Ce document présente une maquette de logiciel de visualisation (appelée Snorky3D en référence à un petit animal aquatique imaginaire) qui utilise un lâcher de particules pour visualiser les champs de vitesse et des champs scalaires issus de calculs de CFD. Cette maquette est optimisée pour permettre une animation fluide et en temps réel sur une station de travail avec de l'ordre du million de particules à partir de maillages d'entrée de quelques millions de mailles. Utilisant le CPU, le moteur d'intégration atteint cinq millions de pas d'intégration par seconde et par coeur de calcul sur les processeurs récents. La maquette permet aussi de réaliser des rendus en différé à très haute définition avec plusieurs dizaines de millions de particules à partir de maillages contenant des dizaines de millions de mailles.

Différentes techniques permettant d'améliorer le rendu visuel et l'inter-activité sont présentées, ainsi que les optimisations utilisées dans le code pour atteindre ce niveau de performances.

La visualisation produite combine les avantages des particules (représentation intuitive de la trajectoire et des vitesses des particules fluides) et du rendu volumique (grâce au très grand nombre de particules utilisé, et à la coloration des particules). La stéréo-vision combinée à cette technique, permet d'appréhender des champs de vitesse in-stationnaires extrêmement complexes.

• *Benoît MATHIEU: CEA-Grenoble, DEN/DANS/DM2S/STMF/LGLS, 38054 Grenoble Cedex*
E-mail: benoit.mathieu@cea.fr.

2 Moteur de calcul et moteur de rendu

Le souhait de conserver un code portable, ainsi que l'empreinte mémoire potentiellement importante des maillages et champs volumiques ont conduit à implémenter le calcul des trajectoires des particules sur CPU et à utiliser le GPU uniquement pour l'affichage en utilisant l'interface OpenGL. Par ailleurs, si une fréquence d'affichage importante est nécessaire à la fluidité de l'animation (plus de trente images par seconde), les trajectoires des particules peuvent souvent être calculées avec un pas de temps plus long, les positions intermédiaires étant calculées par interpolation.

Dans la maquette présentée, le moteur de calcul des trajectoires des particules est implémenté en langage C++ et exécuté en parallèle sur les différents coeurs des CPUs de la machine (parallélisation en mémoire partagée avec pthread). Ce moteur génère des segments de trajectoire qui sont envoyés dans un double tampon de la carte vidéo (un tampon de particules affichées et un tampon de particules en cours de mise à jour). Pour afficher une image, le moteur de rendu interpole linéairement la position des particules sur les segments à l'instant de l'image. Cette opération est réalisée par le GPU et implémentée à l'aide de "shaders" OpenGL. Ainsi, le pas de temps d'affichage est découplé du pas de temps d'intégration des particules et l'affichage des images a un coût CPU quasiment nul, la fréquence d'images étant uniquement déterminée par la puissance de traitement de la carte graphique.

Sur les cartes vidéo NVidia, le transfert des données générées par les CPU sur la carte vidéo est réalisé à l'aide du moteur DMA en utilisant des instructions de copie CUDA et un "pinning" des buffers sur le CPU.

Dans ce cas, le calcul des trajectoires, le transfert des données et l'affichage se déroulent en parallèle et avec un débit maximal des données.

3 Optimisations du moteur de calcul

La trajectoire des particules est décomposée en pas d'intégrations élémentaires, un pas se terminant soit si l'intervalle de temps d'intégration est écoulé, soit si la particule quitte une maille du maillage volumique pour entrer dans une maille voisine.

Pour un pas d'intégration élémentaire, un schéma prédicteur-correcteur est utilisé. Une optimisation du moteur pour augmenter la localité mémoire est décrite. Elle permet permet des gains de performances significatifs (typiquement un facteur trois) mais très variables selon des architectures et les cas tests examinés.

Le traitement des particules est réalisé par plusieurs threads qui traitent chacun plusieurs lots de particules afin de compenser la différence de coût de traitement due au nombre variable de cellules traversées par chaque particule. L'hyperthreading peut apporter un gain de 30%.

L'utilisation d'instructions SIMD permet encore d'augmenter les performances d'un facteur deux environ avec les instructions SSE en simple précision.

Finalement, le moteur d'intégration peut réaliser jusqu'à six millions de pas d'intégration par seconde et par coeur de calcul sur les architectures les plus récentes et sur des maillages tétraédriques.

4 Améliorations du rendu

Plusieurs astuces permettent d'améliorer sensiblement la qualité du rendu visuel des particules par rapport à de simples points ou sphères. Voici les plus importantes :

- coloration des particules en fonction d'un champ scalaire (température, concentration, amplitude du champ de vitesse etc...),
- utilisation de particules semi-transparentes (permet d'augmenter très fortement le nombre de particules tout en conservant une vision en profondeur),
- utilisation de la stéréo-vision (très intéressante conjuguée à la vision en transparence),
- création de particules de tailles aléatoires selon une distribution bien choisie afin de créer une "texture" quand l'image est vue de loin,
- flou de bougé simulé, permettant d'illustrer le champ de vitesse sur des images fixes et améliorant l'impression de continuité du déplacement,
- utilisation du "supersampling" et d'un "tiled rendering" pour le rendu offline à des résolutions arbitrairement élevées,
- rendu à 64 bits par pixel ("high dynamic range") et encodage sRGB pour conserver la luminosité apparente des particules avec le flou de bougé sur les projecteurs et moniteurs calibrés.

5 Interactivité

Les moteurs de calcul et de rendu offrent des options utiles pour explorer l'écoulement.

Il est évidemment possible de tourner autour de l'objet en temps réel, mais aussi de déplacer les sources de particules. Des sources volumiques, surfaciques, filaires, en peigne ou ponctuelles se révèlent utiles.

Les "shaders" openGL permettent d'implémenter très facilement des opérateurs de clipping permettant la réalisation d'effets comme un laser qui éclairerait seulement les particules situées dans un plan. Les plans de coupe peuvent évidemment être déplacés en temps réel.

L'interface du code utilise actuellement des scripts en langage python qui définissent les fichiers contenant les données volumiques, le paramétrage des sources de particules et du moteur de calcul et les paramètres de rendu. Le script peut récupérer les appuis de touches du clavier pour modifier en temps réel les paramètres de calcul ou de rendu. Pour une configuration donnée, on pré-programmera ainsi dans le script les plans de coupe et les sources de particules intéressants, et on prévoira par exemple les déplacements de plans ou de sources pertinents. Cette petite maquette permet ainsi de créer des interfaces graphiques simplifiées mais très bien adaptées à des configurations spécifiques.

6 Conclusion

La puissance de traitement des stations de travail a atteint un niveau suffisant pour permettre d'utiliser un lâcher de particules en temps réel dans les visualisations de calculs de CFD. Avec près de cent millions de pas d'intégration par seconde sur une station de travail bi-processeur récente, on peut maintenant ensemencer l'écoulement finement avec plusieurs millions de particules et observer ces particules en temps réel. Conjuguées à un rendu stéréo-graphique en transparence sur des moniteurs à haute résolution, les visualisations obtenues permettent de comprendre rapidement des simulations d'écoulements particulièrement complexes, puis de créer des films très intuitifs pour illustrer certains aspects de ces écoulements.

La maquette présentée ici a été utilisée pour réaliser plusieurs films présentés sur le mur d'images de la Direction de l'Énergie Nucléaire du CEA à Saclay (voir figures).

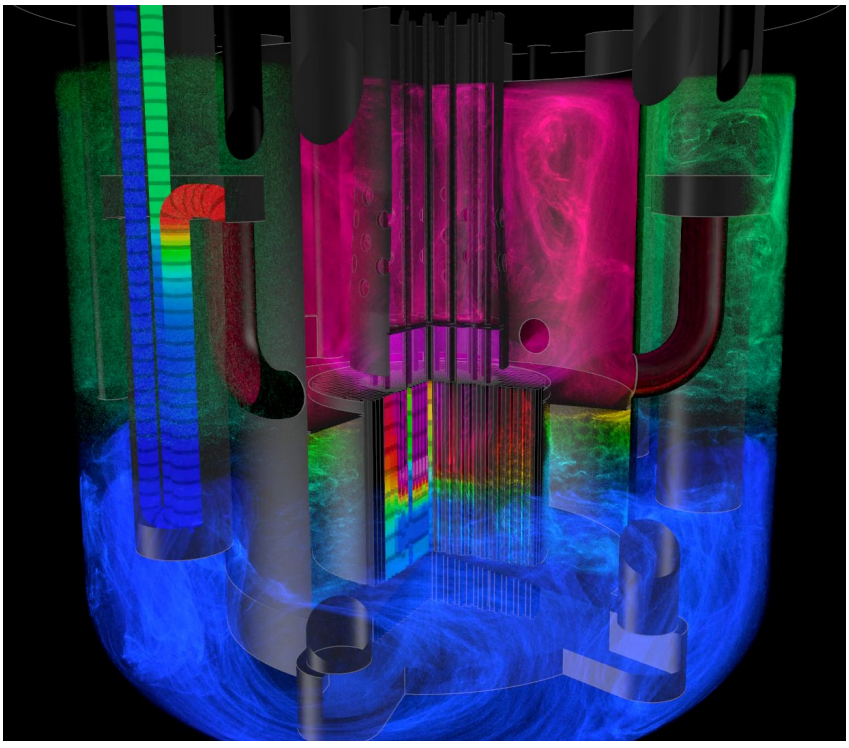

Fig. 1. Extrait d'un film illustrant la thermohydraulique dans le prototype ASTRID, présenté sur le mur d'images à Saclay (DEN/DER/SSTH/LDAL, 2010), 30 millions de particules.

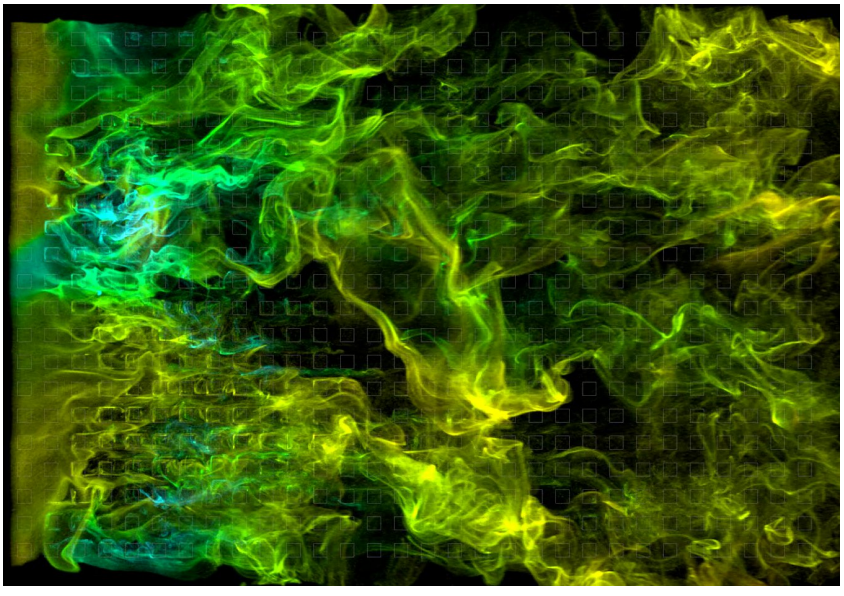

Fig. 2. Film présentant une simulation numérique directe d'un écoulement turbulent (M.Chandesris et A.d'Hueppe, 2011), 20 millions de particules. Le film est visible sur le site web du CINES (http://www.cines.fr/spip.php?rubrique281&lang=fr).