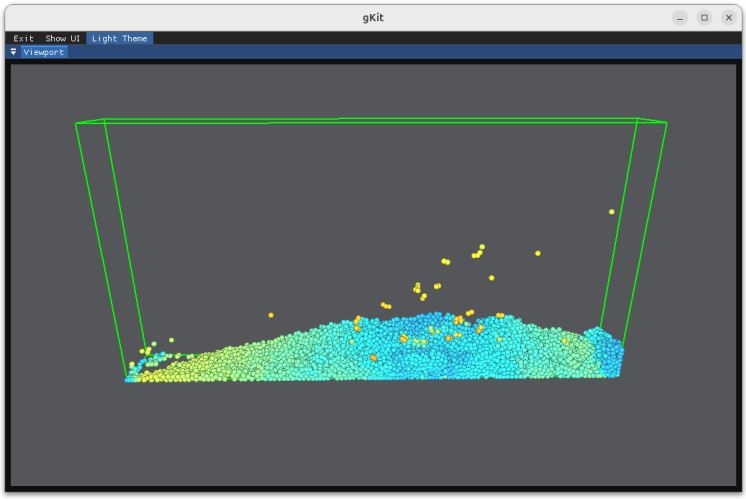
Université Claude Bernard Lyon 1

Rapport Animation en synthèse d'image

Université Claude Bernard Lyon 1

Théo Grillon - p1907354



Master ID3D - IM-TC-ACAMP

Simulation de fluide par méthode SPH

Contents

1 Introduction **2**

2 Modélisation d'un fluide **2**
2.1 Description Eulérienne 2
2.2 Caractérisation d'un fluide 2
2.3 Équations de la dynamique des fluides 2
2.3.1 Première équation 2
2.3.2 Seconde équation 3

3 Méthode SPH **3**
3.1 Modèle de particules 3
3.2 Boucle de simulation 3
3.3 Calcul des densités 4
3.4 Calcul des forces d'intéraction 4
3.5 Gestion des collisions 4

4 Optimisations **5**
4.1 Instanciation 5
4.2 Hachage spatial 5

5 Conclusion **7**

1 Introduction

Dans le cadre de l'unité d'enseignement "Animation, Corps Articulés et Moteurs Physiques" pour le Master ID3D à l'Université Claude Bernard Lyon 1, un travail d'animation par modèle physique a été réalisé. L'objectif était de faire de la simulation de fluide en utilisant la méthode SPH. Dans ce rapport, nous allons aborder les différentes étapes qui ont permis de réaliser cette simulation.

2 Modélisation d'un fluide

Un fluide est un ensemble de particules fluides. La masse d'une particule fluide est constante dans le temps. Cependant, son volume peut varier. On va pouvoir étudier le mouvement d'un fluide en mesurant sa masse volumique ρ, ainsi que sa position, sa pression et sa vitesse.

2.1 Description Eulérienne

À chaque point, on va associer un vecteur vitesse. Cette vitesse est fonction de l'espace et du temps. On va intégrer la vitesse sur l'intervalle de temps considéré pour obtenir la trajectoire de la particule.

2.2 Caractérisation d'un fluide

Un fluide se caractérise par un **champs de vitesse v**, une **densité volumique** ρ et un **champs de pression p** : force par unité de surface que le fluide exerce sur n'importe quoi.

2.3 Équations de la dynamique des fluides

La simulation consiste à calculer l'évolution de ces quantités au cours du temps à partir de deux équations.

2.3.1 Première équation

La première équation assure la conservation de la masse. La masse du domaine reste constante pendant son mouvement :

$$\frac{d}{dt}m = \frac{d}{dt}\int_{D_t} \rho \; dv = 0$$

En utilisant les formules de dérivées particulaires d'une intégrale de volume, on obtient ceci :

$$\int_{D_t} (\frac{\partial}{\partial t}\rho + \rho\nabla.\vec{v}) \; dv = 0$$

On en déduit les formes locales de la conservation de la masse :

$$\frac{\partial}{\partial t}\rho + \rho\nabla.\vec{v} = 0$$

2.3.2 Seconde équation

La seconde équation est issue de l'équation du mouvement. Seconde équation de Newton :

$$\mathbf{f} = m\mathbf{a} \implies \mathbf{f}/V = m/V\mathbf{a} = \mathbf{f}/V = \rho\mathbf{a}$$

Nous avons ainsi pour le fluide :

$$\rho\frac{D\mathbf{v}}{Dt} = \mathbf{f}$$

3 Méthode SPH

Pour simuler un fluide, on utilise la méthode **SPH** pour **S**moothed **P**article **H**ydrodynamics présenté par Matthias Müller en 2003.

On représente le fluide par l'ensemble de paramètre suivant :

- Densité : ρ_0 en kg/m^3
- Module de Bulk : K en GPa
- Viscosité : μ en $Pa.s$

3.1 Modèle de particules

On représente le fluide par un ensemble de particules. Chaque particule a une masse m et on va également définir un rayon d'interaction h. Chaque région du fluide est représentée avec une particule i par sa position r_i, sa vitesse v_i et sa densité ρ_i. Les particules vont interagir entre elles afin de simuler la dynamique des fluides. Pour ce faire, on va devoir calculer leur densité ρ_i et des forces d'interaction f_{ij}, avec j les indices des particules présentes dans le rayon h de la particule i.

3.2 Boucle de simulation

La boucle de simulation se divise en trois étapes :

1 Calcul des forces appliquées aux particules :

  1.1 Force d'interaction entre particules : $\mathbf{f}_{ij}^{interact}(t)$

  1.2 Force de gravité : $\mathbf{f}_g = m\mathbf{g}$

2 Calcul des accélérations des particules :

  2.1 Principe fondamental de la dynamique :

$$\sum \mathbf{F}_i = m\mathbf{a}_i = \mathbf{a}_i \implies \mathbf{a}_i(t) = \sum \mathbf{F}_i(t)/m = \frac{1}{\rho_i}\sum_{j\in N_i}\mathbf{f}_{ij}^{interact}(t) + \mathbf{g}$$

3 Intégration pour obtenir les nouvelles vitesses et positions :

3.1 $\mathbf{v}_i(t+dt) = \mathbf{v}_i(t) + dt\mathbf{a}_i(t)$

3.2 $\mathbf{x}_i(t+dt) = \mathbf{x}_i(t) + dt\mathbf{v}_i(t+dt)$

En résumé, pour calculer les accélérations, on doit d'abord calculer les forces, donc les forces d'interaction. Pour les calculer, on doit d'abord calculer la densité de chaque particule.

3.3 Calcul des densités

Pour calculer la densité de chaque particule, on utilise la formule suivante :

$$\rho_i = \frac{4m}{\pi h^8} \sum_{j \in N_i} (h - r)^3$$

On va rechercher les voisins j de la particule i. Le plus simple est de vérifier la condition suivante : $h - r > 0$ où $r = ||\mathbf{r}_i - \mathbf{r}_j||$ la distance au carré entre les particules i et j. L'optimisation dont on parlera dans la dernière section de ce rapport permet de ne parcourir qu'un sous-ensemble de particules se trouvant dans le voisinage de la particule i.

3.4 Calcul des forces d'intéraction

Une fois que la densité de chaque particule est calculée, on peut calculer les forces d'interaction. On utilise pour cela la formule suivante :

$$\mathbf{f}_{ij}^{interact} = \frac{m_j}{\pi h^4 \rho_j}(1 - q_{ij})[15K(\rho_i + \rho_j - 2\rho_0)\frac{(1 - q_{ij})}{q_{ij}}\mathbf{r}_{ij} - 40\mu\mathbf{v}_{ij}]$$

avec :

- $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$
- $\mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j$
- $q_{ij} = ||\mathbf{r}_{ij}||/h$

On calcule les forces d'interaction appliquées sur la particule i et la particule j simultanément. La force appliquée sur j est simplement l'inverse de celle appliquée sur i : $\mathbf{f}_{ij}^{interact}(t) = -\mathbf{f}_{ji}^{interact}(t)$

3.5 Gestion des collisions

Pour gérer les collisions des particules avec une boîte, on va la définir par deux points min et max. À chaque itération de la boucle de simulation, on vérifie pour chaque particule si elle se situe à l'intérieur de la boîte (comparaison sur chaque axe par rapport à min et max). Dans le cas où une particule sort de la boîte, on la replace à l'intérieur et on calcule la nouvelle vitesse en prenant la direction $\mathbf{reflect} = V_i - 2 * V_i \cdot \mathbf{n} * \mathbf{n}$ avec $\mathbf{n}$ la normale au point de contact.

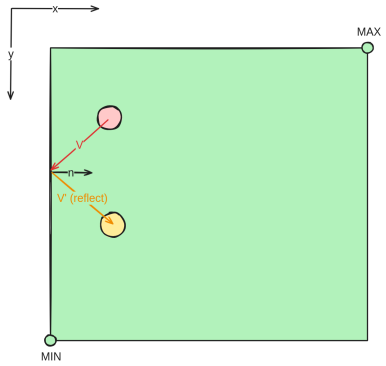


Figure 1: Reflect

4 Optimisations

Dans cette section, nous allons aborder les optimisations apportées au code de base. L'instanciation d'objet à été utilisée pour accélérer la partie rendu. Pour optimiser la partie calcul, je me suis inspiré de cette vidéo Youtube, dans laquelle la méthode de hachage spatiale est présentée.

4.1 Instanciation

Le code de base effectue un appel à *draw* pour chaque particule. Un appel à cette fonction engendre une communication entre le CPU et le GPU qui est quelque peu coûteuse. On va vouloir réduire le plus possible les interactions entre CPU et GPU.

Un moyen simple est de faire de l'instanciation d'objet. Chaque particule est représentée par une sphère. Lors de l'initialisation de l'application, on crée un **VBO** (**V**ertex **B**uffer **O**bject) pour contenir la géométrie de cette sphère et un **IBO** (**I**ndex **B**uffer **O**bject) pour sa topologie. On va également avoir besoin de passer au `vertex shader` un tableau de translations correspondant aux positions de chaque particule. Cela va permettre d'avoir des positions différentes pour chaque instance. On n'a plus qu'à appeler `glDrawElementsInstanced` en fournissant le nombre d'instances que l'on souhaite pour dessiner toutes les particules en un seul `draw call`.

4.2 Hachage spatial

Le hachage spatial consiste à découper l'espace en une grille et à définir une valeur de hachage pour définir la cellule dans laquelle chaque particule se trouve.

On définit un premier tableau `_spatial_lookup` qui va contenir la paire suivante : l'`indice de la particule` et la `clé de cellule` qui va être cal-

culée à partir de la position de la cellule contenant la particule et d'une fonction de hachage. Pour remplir ce tableau, on parcours l'ensemble des particules. Pour chacune, on calcule leur position dans la grille 3D. Cela va nous donner un vecteur de trois valeurs à partir duquel on calcule une valeur de hachage. Ensuite, on applique un modulo sur la valeur de hachage afin de s'assurer que la clé soit comprise dans le tableau de dimension correspondante au nombre de particules. Une fois ce tableau rempli, on le trie par ordre croissant (ou décroissant) afin de regrouper les éléments ayant la même clé.

Un second tableau `_start_indices`, va stocker l'indice du premier élément d'un ensemble de particules regroupées dans une même cellule (ensemble de particules ayant la même clé). Cela va nous permettre de restreindre le nombre de particules parcourues lors des calculs de densité et de force d'interaction. En effet, lors de ces calculs, il suffit de calculer la clé de la particule courante et de récupérer l'indice dans `_start_indices` du premier élément dans le tableau `_spatial_lookup` ayant la même clé. Puis, d'itérer sur la suite du tableau `_spatial_lookup` tant que la clé est identique. Il se peut que certaines particules qui ne sont pas réellement présentes dans la même cellule que la particule courante aient la même clé, mais cela n'est pas très grave et permet tout de même de réduire considérablement le nombre de particules parcourues.

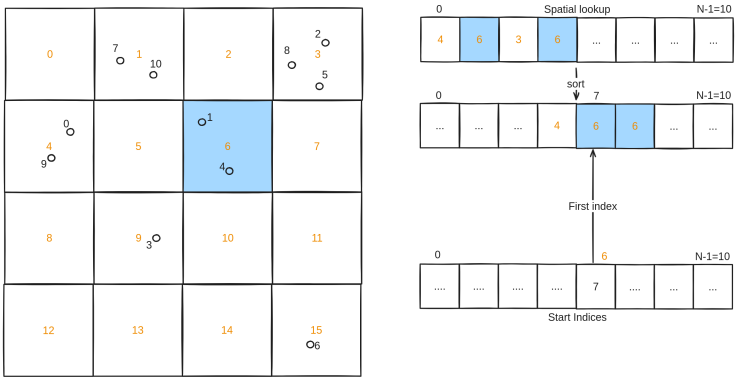


Figure 2: Hachage Spatial

Avec l'utilisation de cette technique, on peut afficher plus de 10000 particules sans problème, là où ça devenait compliqué au-dessus de 3000).

5 Conclusion

Une interface graphique a été mise en place pour pouvoir modifier les paramètres de la simulation comme la viscosité, le module de Bulk ou encore les dimensions de la boîte de collision. Cela permet de trouver les valeurs qui donnent une simulation de fluide correcte. On peut également afficher la densité de chaque particule ainsi que leur vitesse.

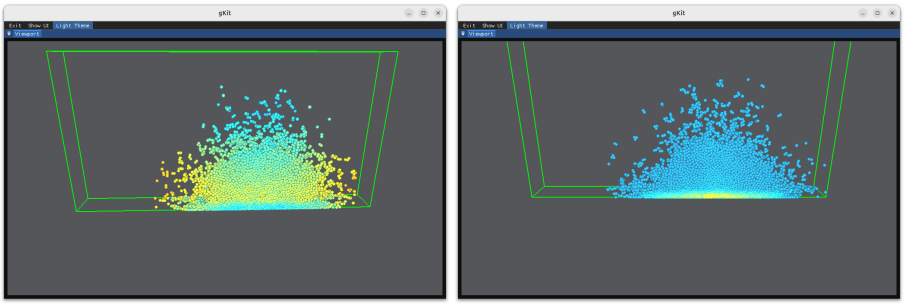


Figure 3: Vitesse vs Densité

Dans l'ensemble, je suis plutôt satisfait du résultat final et des optimisations apportées afin d'avoir la possibilité d'afficher et de gérer plus de particules. Même si des améliorations pourraient être faites en approfondissant le sujet.