TP/Projet Informatique Graphique

Vayssade Jehan-Antoine

petit soucis d'hebergement qui m'a empecher de finir certaines sections ce soire
le minimum demander est tout de même fait (3 sujet)

ce seras pour la v2 !

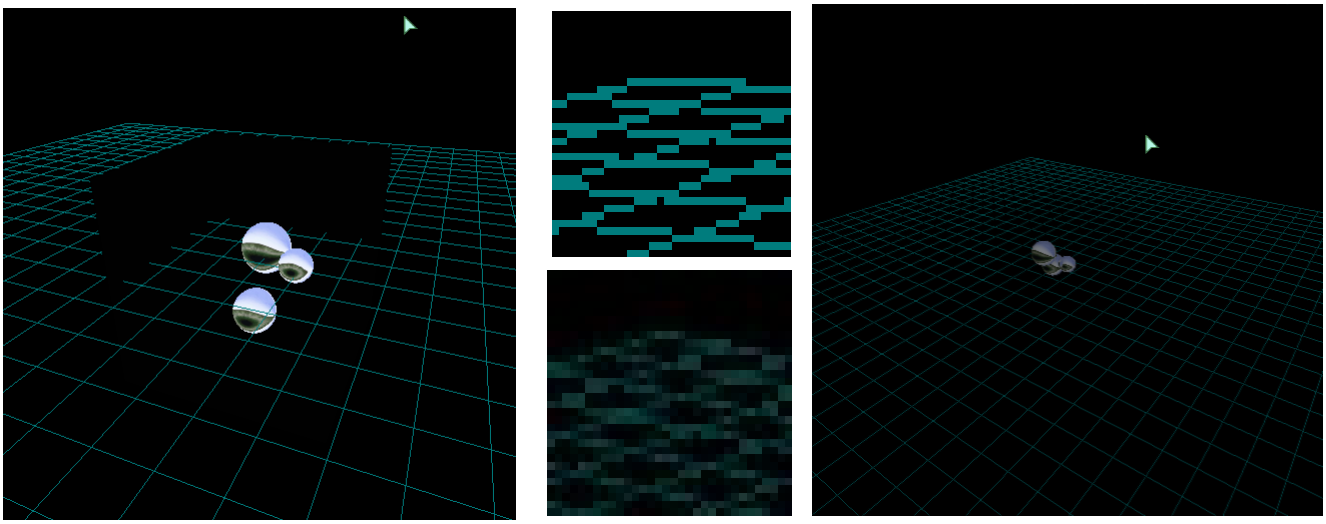
Rendu

- ~~Order indepedent transparency~~
- Antialiasing FXAA (fait)
- Redux (pour le rendue hdr) (fait)
- Tonemapping (en cours) + bloom (fait)
- Deffered Shading (en cours)
- SSAO (fait)

Géométrie

- Subdivision catmull clark (en cour)
- Box-Spline
- ~~Simplification~~
- Visualisation de surface implicite (fait)

Antialiasing FXAA



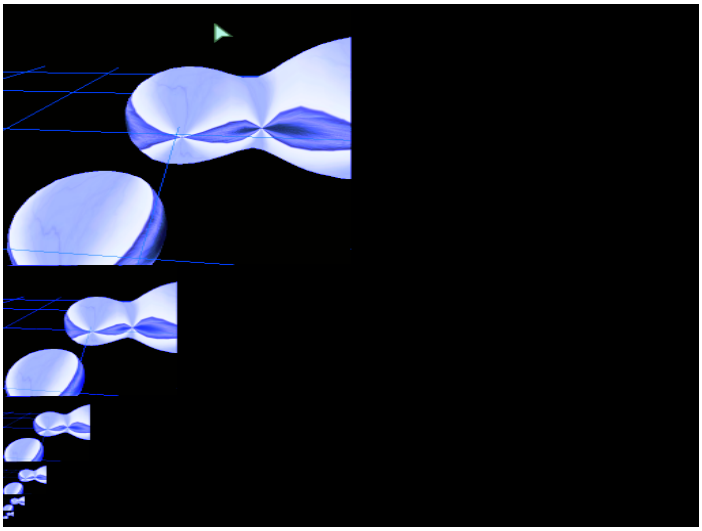
FXAA fonctionel, quelques cliniotement lorsque les lignes bouges, ainssi que quelques effects de distortions au jonctions de pixels, c’est le resultat attendue de la méthode, même s’il n’est pas parfait.

C’est d’autemp plus visible lorsqu’il y a une grande différences de couleur (ie l’arriere plan)

Cette méthode est baser sur la difference de luminausiter entre les pixels voisins, cette différence détermine alors un offset de texture pour trouver la couleur du texel, mais aussi pour influer sont intensiter un genre de “flouetage”

Libreoffice n’aide pas à bien apprécier la qualiter du FXAA

Redux (pour le rendue hdr)



Parfaitement fonctionel
bonne performance entre glViewPort et le discard des zones inutiles
tout est calculer sur une seul texture

Tonemapping+bloom (en cours)

Les méthode de calcules suivantes ont été integer, il manque la jonction entre le redux, le tonemap et le bloom, mon système de multi-pass sommaire fait défaut.

vec3 linearToneMapping(vec3 color, float gamma)

vec3 simpleReinhardToneMapping(vec3 color, float gamma)

vec3 lumaBasedReinhardToneMapping(vec3 color, float gamma)

vec3 whitePreservingLumaBasedReinhardToneMapping(vec3 color, float gamma)

vec3 RomBinDaHouseToneMapping(vec3 color, float gamma)

vec3 filmicToneMapping(vec3 color, float gamma)

vec3 Uncharted2ToneMapping(vec3 color, float gamma)

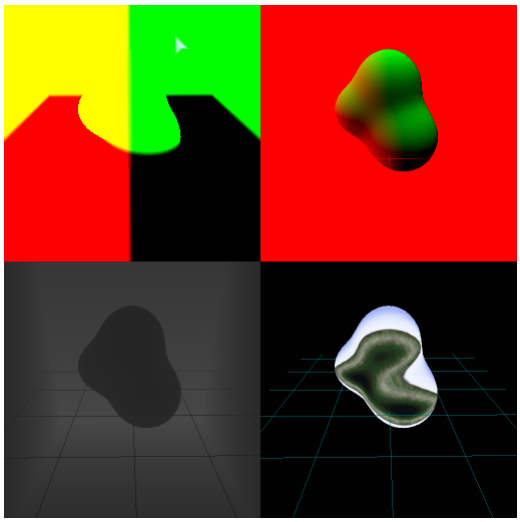
Deffered Shading (en cours)

La gestion du gbuffer est operationel

J’ai adapter mon blinn-phong pour utiliser les informations du gbuffer.

Il me reste juste a envoyer les lumières au shaders et à trouver la bonne equation pour lire le “gbuffer.position”

Pas de resultat pour le moment



SSAO

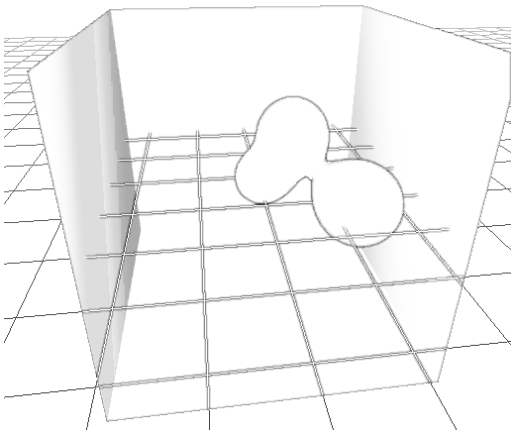


Illustration 1: 1 iteration

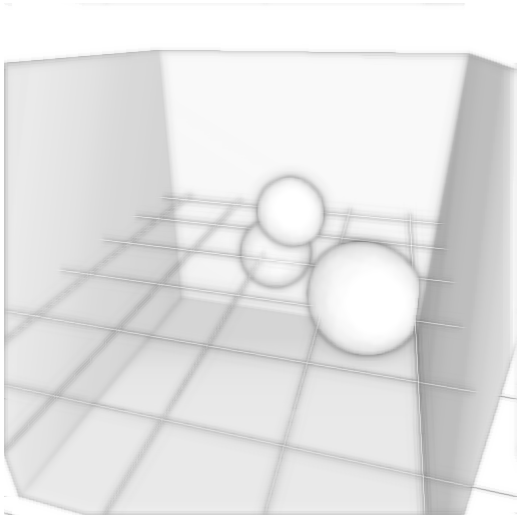


Illustration 2: 4 iteration + blur

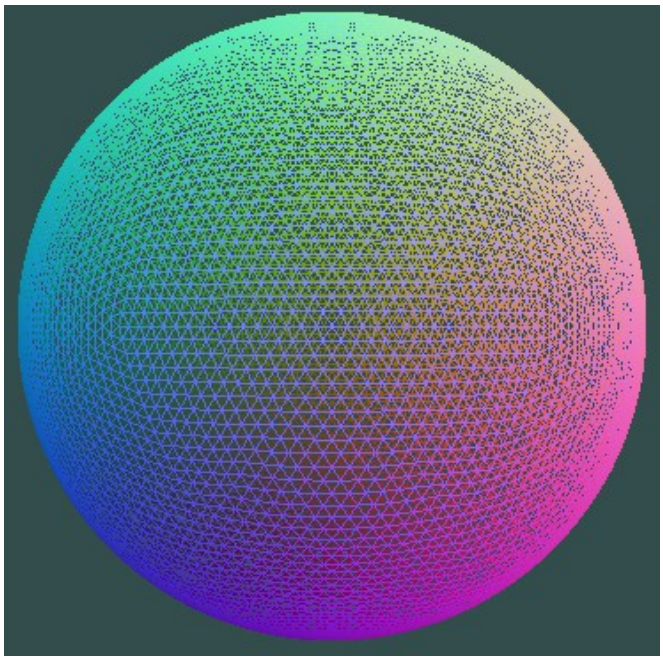
Petite méthode perso, simple et plutot performante même si elle nécéssite une recherche de depth entre 4 et 10 pixel voisin avec un pass de blur pour lisser les differences de depths et optinir un jolie rendue.

Sinon des lignes de “ruptures” peuvent apparaitre sur les plans, visible sur l’Illustration de gauche.

Grâce au blur, il est aussi possible d’améliorer le SSAO en omettant le calcule de certain pixel, la pass de blur est donc la pour lisser l’erreur. Je n’ai pas fait cette dernière optimisation.

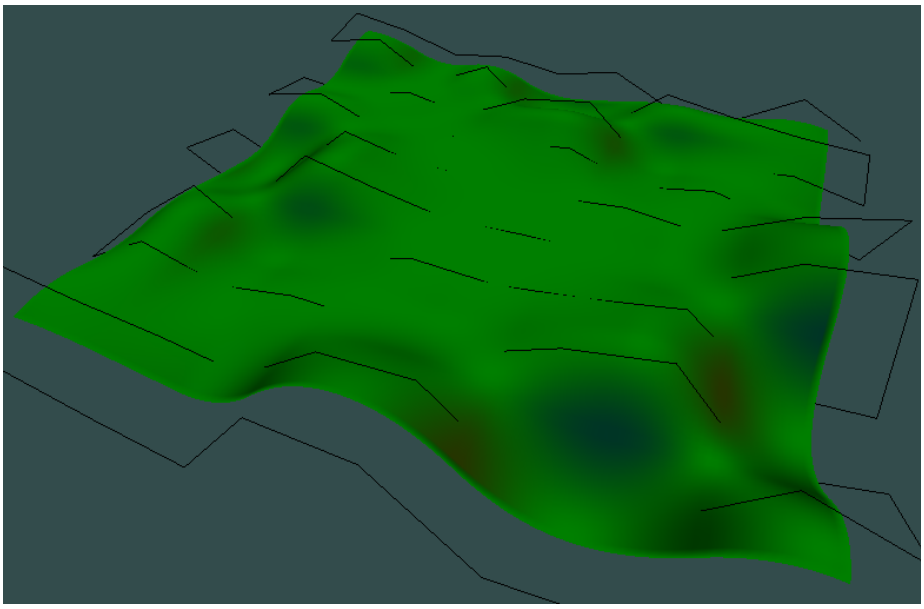
Subdivision (catmull clark)

En cour d’élaboration, la subdivision de la topologie est géré, cependant je n’ai pas fini l’interpolation des vertices.



B-Spline

Dans le cadre du TP de loic Barth, ce “moteur” integre aussi les box-splines.



Visualisation de surface implicite

J’ai utiliser la méthode d’écrite sur le site de paulbourke.net, les différents exemples précédent montre la visualisation d’une métaball. J’ai envisager plusieurs approche d’optimisation, mais aucune de très probant sans passer des jours a developpez.

J’ai donc simplement minimiser le nombre de donner envoyer sur le GPU par rapport à la méthode donner, cette petite optimisation m’a permit d’obtenir un rendue temps réel ~20% plus éffisiant, on pourrais aussi optimiser la table des triangles et des arrêtes pour gagner encore un peux de bande-passante.

Aller plus loin reviendrais a mettre en place un structure en sprase octree afin de supprimmer les calcules de distance des metaballs dans certaines sections, ou avec une prévision par rapport au déplacement de chacune d’entres elles (intérieur)

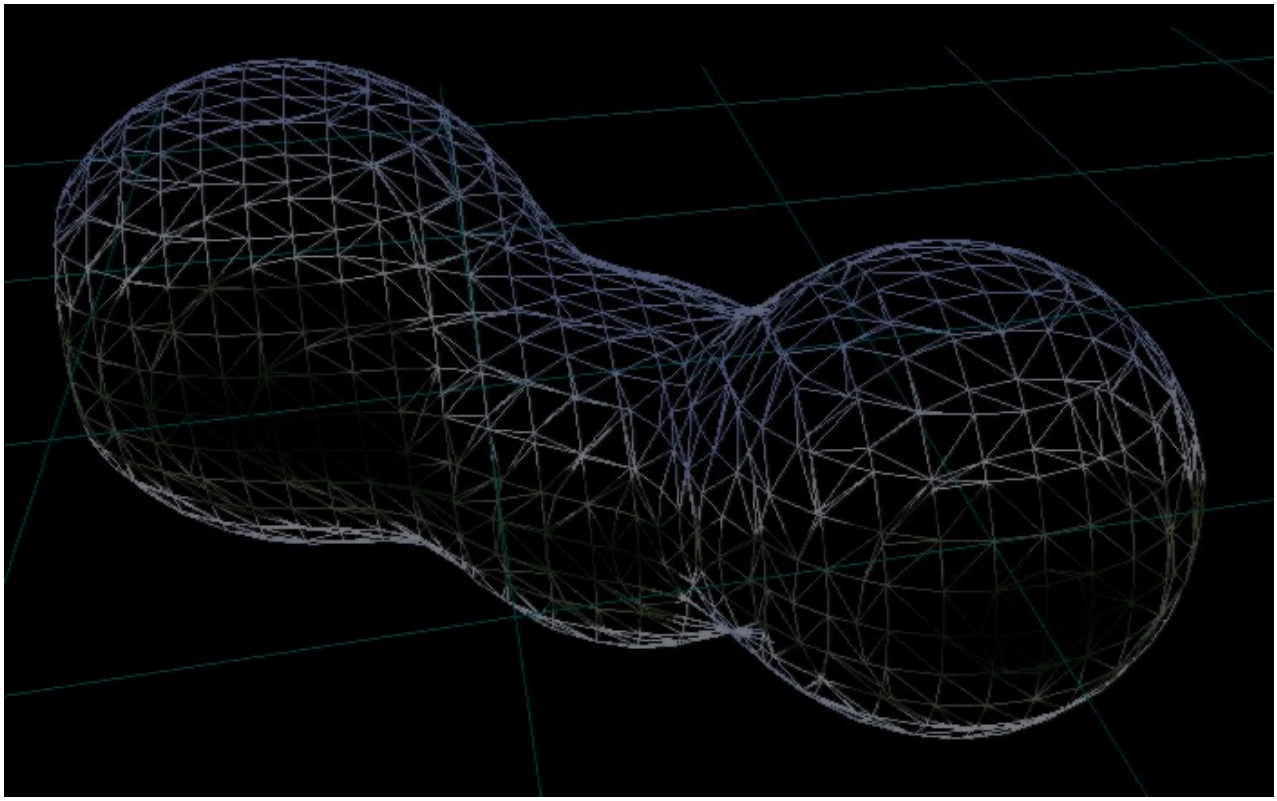


Illustration 3: trois Metaball avec interpolation dans une grille de 40x40x40 et texturer