Compte Rendu - TP ETI5-IMI
Shaders avancés et Marching Cubes

Di Folco Maxime - Girot Charly

27/10/2017

1 Parallax Mapping - Donner l'illusion du relief

1.1 Utilisation des textures de normales

Avant : utilisation de textures *colorées* appliquées directement sur un maillage. Exemple : dessiner un mur de briques comme sur la Fig.1.1

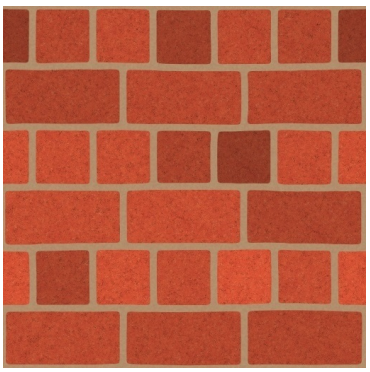
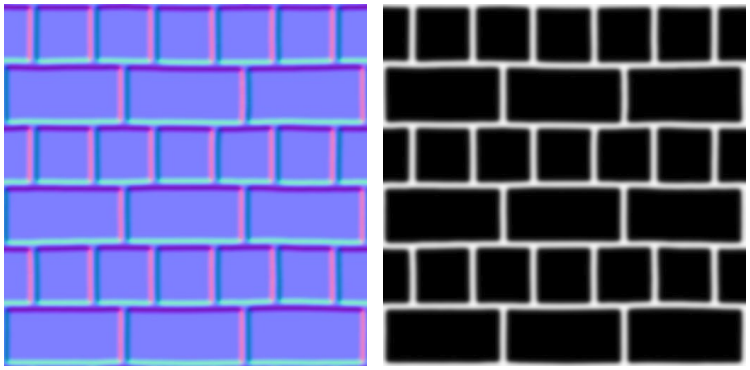


FIGURE 1 – Images à reconstruire sous la forme d'un panorama

Désormais : utilisation des textures pour d'autres informations et utilisations (modifications de normales Fig2(a), stockage d'informations de profondeurs Fig2(b)).



(a) (b)

FIGURE 2 – Textures contenant les informations de normales (a) et de profondeur (b)

Q1
Première implémentation : Parallax mapping implementé avec la carte des normales, comment est ce que la lumière interagit avec la surface : La lumière est fixe mais les ombres évoluent en fonction de la rotation de l'objet. Il est possible ed distinguer un relief entre les briques et leur support (le mur). Cela provient du fait que les normales ne sont plus seulement orientés vers nous selon l'axe z mais sont désormais liées aux axes x,y et z de la surface en fonction de la carte des normales representée Fig2(a) où la composante rouge indique une modification de la normale selon l'axe x, verte pour l'axe y et bleue pour l'axe z. Les normales ne sont ainsi plus toutes orientées dans le même sens selon la surface de l'objet, mais orientées différement selon chaque fragment constituant notre objet. On obtient ainsi grace à l'illumination des surfaces beaucoup plus détaillés.

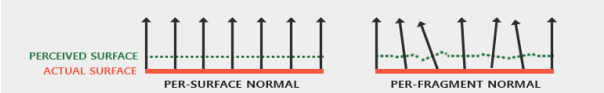


FIGURE 3 – Principe de l'orientation des normales en parallax mapping - tiré de : https ://learnopengl.com/# !Advanced-Lighting/Normal-Mapping

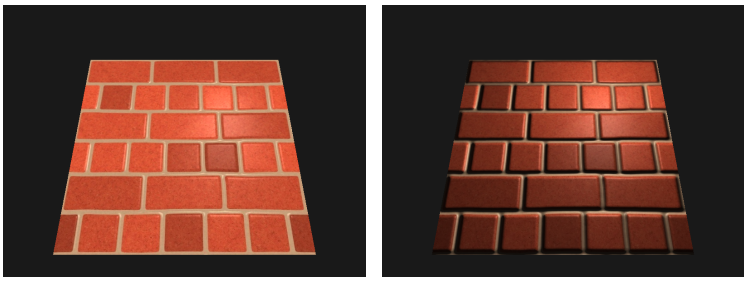
Q2
fichier main.cpp, calcul des tangeantes et bitangeantes, comment simplifier ? Les tangeantes bi-tangeantes sont calculées dans l'espace TBN (voir normal mapping pour ajouter des détails). Pour simplifier on aurait pu utiliser un simple normal mapping en utilisant la texture des normals en orientant simplement ces dernières selon la map mais cela pose le problème de ... pas tout compris au problème sur le site sauf que ca marche pas tout le temps en fonction de ... fatigue.

Comment sont gérées les différentes textures ? Rôle de glActiveTexture(). Chaque texture est stockée dans un indice GLTEXTUREn puis on lit le buffer courant activée à l'aide de glActiveTexture(GLTEXTUREn) à l'aide de glbindtexture() ;

Q3
Rôle de chaque variable dans les shaders Vertex Shader in vec3 position ; Sommet habituels in vec3 normal ; Normal données in vec2 tex_coords ; Coordonées de texture in vec3 tangent ; Tangeante et bitangeante calculés précédemment in vec3 bitangent ;

On calcule la matrice TBN out vec3 vf_frag_pos = model * position ; out vec2 vf_tex_coords ; out vec3 $vf_tangent_light_pos$ = TBN * $light_pos$; out vec3 $vf_tangent_view_pos$ = TBN * $view_pos$; out vec3 $vf_tangent_frag_pos$ = TBN * vf_frag_pos ;

Fragment Shader : en fonction de la direction de la lumière, de la vue et des fragments tangeants on peut calculer les directions de la vue et de la lumière En fonction des différentes textures envoyées (couleur (diffuse) et normales) et des coordonnées de texture, on peut calculer la couleur des fragments. Les variables vf sont exprimées dans l'espace TBN (donc liées à la surface de l'objet). On utilise pas height tex car on fait du normal mapping pas du parallax mapping (pas encore).



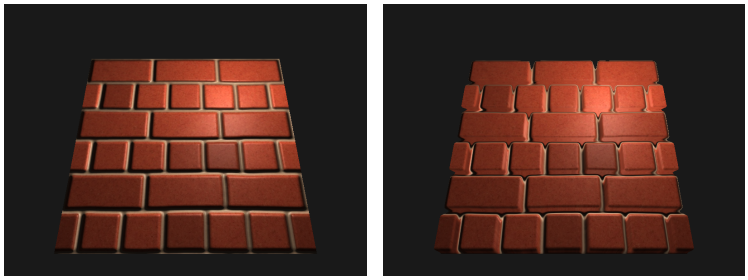
(a) (b)

FIGURE 4 – Résultat de l'application de texture couleur (a) auquel on applique la gestion des normales (b)

1.2 Ajout d'un effet de profondeur avec l'utilisation des cartes de hauteur

Modification des coordonnées de texture en fonction de la carte d'élévation et de la position de la vue. Explicqtion géométrique de ce qu'on fait pour que donner cet effet de profondeur et du pourquoi ça marche mieux quand on a une vue rasante.

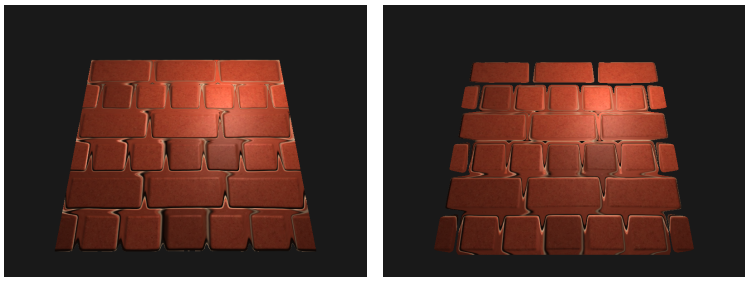
On applique ensuite l'algorithme suivant pour ajouter l'effet de profondeur :



(a) (b)

FIGURE 5 – Comparaison de la gestion des normales (a) auquel on applique une carte de hauteur (b)

Critique des limitations du parallax mapping.



(a) (b)

FIGURE 6 – Résultats de la carte des profondeurs pour un facteur de hauteur trop faible (a) trop haut (b)

2 Marching Cubes

C'est quoi l'algo des marchings cubes

Question 5 : Repérez quel groupe de shaders et quelle partie du code C++ sont liés à chaque étape. Essayez de faire un schéma indiquant comment les différentes étapes echangent des données via des FBO ou des TF.

Question 6 : Lisez la documentation de la fonction glDrawArraysInstanced , que réalise cette fonction ? Aurait-on pu s'en passer ? Quelle est l'utilisation plus classique de cette fonction ?

Question 7 : Comment s'utilise la fonction glActiveTexture ? Déjà demandé son rôle plus haut, peut être répondre à tout en même temps

Question 8 : Dans les shaders, on trouve des variables de type isampler2D et d'autres de type sampler2D. Quelle est la différence entre ces deux types ? (c'est simplement entier ou non ?) Que réalise la fonction texelFetch ?

Question 9 : Corrigez le calcul des normales en supposant que sa direction est donnée par le gradient de la fonction de densité. Quel fichier avez-vous modifié pour cela ?

L'aspect crénelé de la sphère vient du fait que la position des vertex est toujours choisie au milieu des arètes des cubes. Une meilleure approche consiste à réaliser une interpolation linéraire pour choisir cette position. Question 10 : Réalisez cette interpolation de manière à obtenir une sphère bien lisse comme celle de la figure 9.