

TP09 : Projet de Synthèse d'Images

Génération de terrain aléatoire

Line POUVARET, Mickaël TURNEL, Antoine THEBAUD

2015-2016

1 Présentation de notre projet

Nous avons réalisé un programme de génération de terrain qui utilise un bruit de Perlin pour créer un champ de hauteur.

Voici l'ordre détaillé des étapes que nous réalisons.

1.1 Première passe : génération du bruit

Elle consiste à générer le bruit grâce à un shader approprié (noise.vert, noise.frag).

On envoie au vertex shader également le vecteur de mouvement qui va nous permettre de modifier la map de bruit générée en fonction de celui-ci.

Le bruit ainsi généré est produit dans une texture de la taille de la grille du terrain que l'on dessine dans un fbo.

1.2 Deuxième passe : normales

Celle-ci sert à générer la map des normales des sommets en utilisant d'autres shaders (normal.vert, normal.frag).

Le fragment shader se sert bien évidemment de la map de hauteur générée précédemment.

Encore une fois, elle est générée dans une texture dessinée dans le même fbo.

1.3 Troisième passe : shadow map

Après cela, nous avons une passe à part pour générer la shadow map de notre rendu dessinée dans un autre fbo (shadow-map.vert, shadow-map.frag).

Dans le vertex shader, nous mettons à jour la position des sommets selon l'espace de lumière (matrice de modèle-vue-projection) et on ne fait rien dans le fragment shader.

Nous utilisons des soft shadows avec la Loi de Poisson pour générer nos ombres.

1.4 Quatrième passe : rendu

Enfin, on rend notre scène en réalisant un displacement mapping selon le champ de hauteur généré. (rendering.vert)

1.5 Cinquième passe : post-process

On ajoute un Phong shading dans le fragment shader pour la lumière (en utilisant la normal map) ainsi que nos ombres projetées grâce à la shadowmap. (rendering.frag)

Toujours dans le fragment shader du rendu, on ajoute un effet de brouillard blanc sur la scène en fonction de la profondeur par rapport à la position de la caméra.

Et pour finir, nous utilisons des textures 2D qui nous permettent de déterminer la couleur d'un fragment selon la hauteur.

2 Fonctionnalités du programme

Les paramètres du bruit sont les suivants :

- amplitude : 50.0
- fréquence : 2.0
- persistance : 0.4
- nombres d'octaves : 10.0

Il y a la possibilité de "naviguer" dans le terrain en appuyant sur la touche Z pour avancer et S pour reculer.

Le mouvement du terrain suit le mouvement de la camera si on laisse le clic gauche appuyé.

Il est également possible de "descendre" et "monter" le terrain sachant qu'il n'a qu'une limite en descente (au niveau de l'eau) avec les touches Q et D (Q pour descendre, D pour monter).

Le rendu de lumière, les ombres projetées et le brouillard peuvent être activées/désactivées individuellement.

Nous utilisons une texture 2D qui nous sert de colormap sur le terrain.

Chaque pixel de sommet recevra le pixel de la texture qui lui sera associé grâce à sa hauteur.

Il y a aussi la possibilité d'utiliser plusieurs textures individuelles qui seront appliquées selon la hauteur du sommet.

On peut aussi afficher les différentes map du programme à savoir :

- la height map
- la shadow map
- la normal map

Celles-ci changent bien évidemment lorsque l'on se déplace avec les touches Z et S.

2.1 Description des raccourcis clavier

- Z : avancer
- S : reculer
- Q : descendre
- D : monter
- W : afficher/masquer la shadow map

- C : afficher/masquer la normal map
- X : afficher/masquer la height map
- L : activer/désactiver le rendu de lumière
- G : activer/désactiver les ombres
- B : activer/désactiver le brouillard
- F : afficher les fps
- T : changer de mode de texture (entre la texture 2D colormap et les textures individuelles)
- I : réinitialiser

2.2 Description des raccourcis souris

- clic gauche enfoncé : permet de bouger l'angle de la caméra
- clic droit : permet de modifier la position de la source de la lumière

3 Captures d'écran

3.1 Au lancement du programme

Voici la capture d'écran obtenue au lancement du programme :

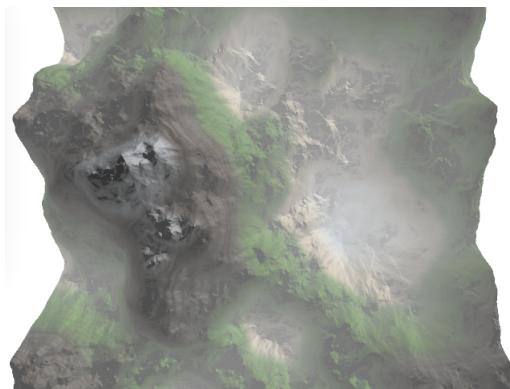


FIGURE 1 – Vue du terrain au lancement

La vue par défaut est par dessus le terrain.

En laissant le clic gauche enfoncé et en faisant un mouvement vers le haut, on obtient cette vue :

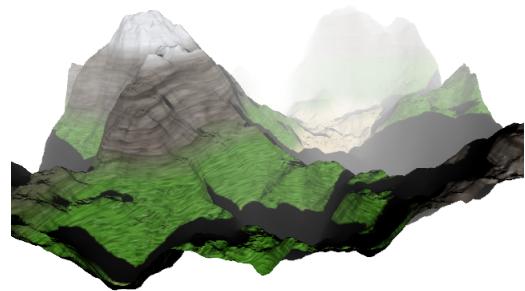


FIGURE 2 – Vue du terrain de face

3.2 Activer/désactiver des post-process

On observe que nous avons par défaut les ombres, la lumière et le brouillard. Si l'on désactive le brouillard, on obtient cette vue :

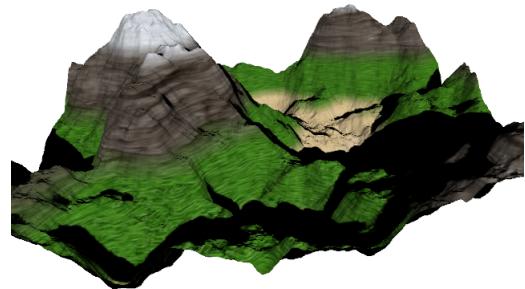


FIGURE 3 – Terrain sans brouillard

Si l'on désactive uniquement les ombres, on obtient cette vue :

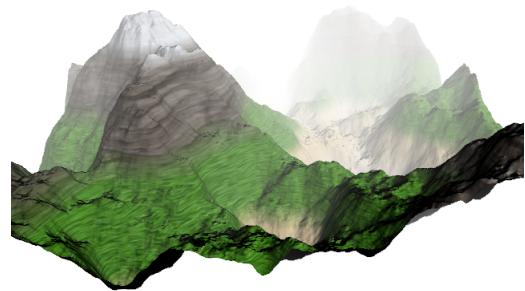


FIGURE 4 – Terrain sans ombres

Si l'on désactive uniquement la lumière, on obtient cette vue :

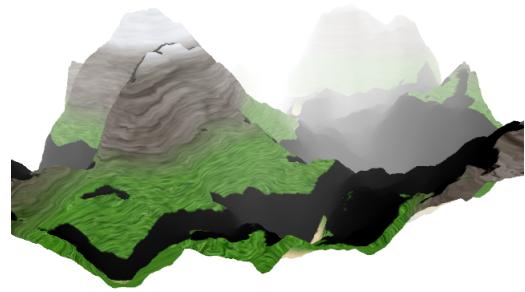


FIGURE 5 – Terrain sans lumière

Et enfin, si on désactive les trois, on obtient cette vue (beaucoup moins agréable à regarder) :

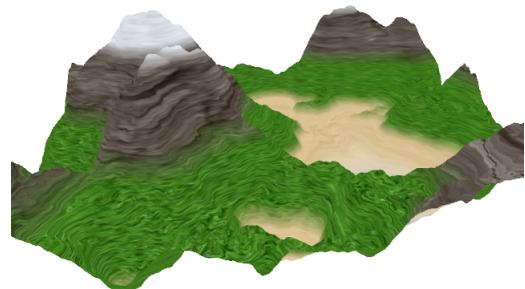


FIGURE 6 – Terrain sans ombres, brouillard, lumière

3.3 Afficher/Masquer les map

3.3.1 La height map

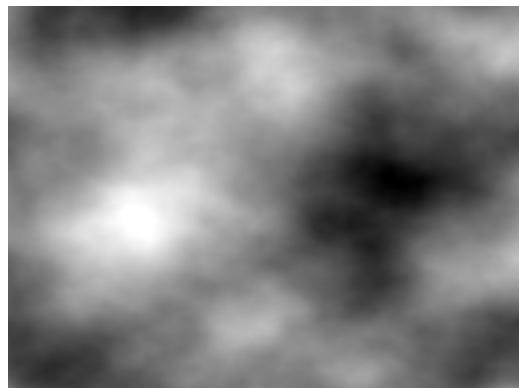


FIGURE 7 – height map

3.3.2 La normal map

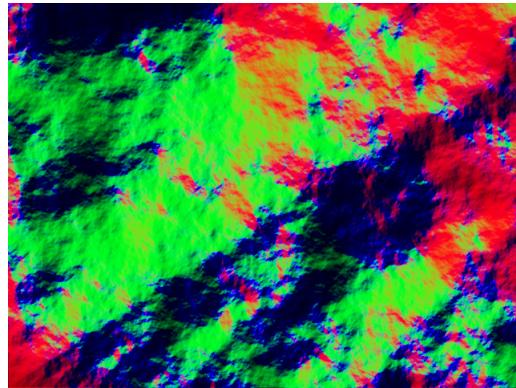


FIGURE 8 – normal map

3.3.3 La shadow map



FIGURE 9 – shadow map

3.4 Changer le mode de texture

Voici l'affichage d'une scène de notre terrain avec la texture 2D de type colormap :

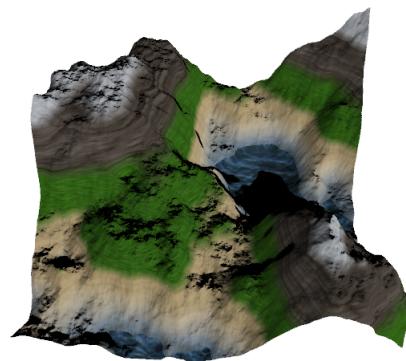


FIGURE 10 – Scène avec texture 2D colormap

Et voici l'affichage de la même scène avec les textures 2D individuelles :



FIGURE 11 – Scène avec textures 2D individuelles

4 Textures

4.0.1 Texture 1D colormap

Nous utilisions cette texture avant de passer en texture 2D.



FIGURE 12 – colormap 1D

4.0.2 Texture 2D colormap

Nous utilisions aussi cette texture avant d'en utiliser une deuxième version.



FIGURE 13 – colormap 2D

4.0.3 Texture 2D colormap v2

Voici la version utilisée actuellement de texture colormap :



FIGURE 14 – colormap 2D v2

4.0.4 Textures individuelles



FIGURE 15 – Neige



FIGURE 16 – Roche



FIGURE 17 – Herbe



FIGURE 18 – Sable

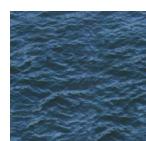


FIGURE 19 – Eau

5 Problèmes et améliorations

5.1 Problèmes

5.1.1 Textures

Nous avons eu un soucis dans un premier temps avec la texture 2D colormap car pour les petites et grandes valeurs de hauteur, la texture "clamp", ce qui n'était visuellement pas très beau.

Nous avons remédié au problème en forçant à choisir les textures individuelles pour ces grandes et petites valeurs en les mettant en mode GL_REPEAT.

Il y a également un soucis (toujours présent), pas des plus gênants mais visible tout de même avec le mouvement.

En effet, lorsque l'on avance ou recule, on constate que les textures ont tendance à bouger le long du relief de façon horizontale.

Cela s'explique par le fait que nous utilisons les coordonnées du sommet pour accéder aux coordonnées dans la texture.

Or, les coordonnées du sommet changent en fonction du mouvement.

Nous n'avons pas eu le temps de nous pencher sur la question afin de trouver une solution convenable à ce problème.

Nous utilisions une texture 1D colormap au départ, et avec celle-ci, nous n'avions bien évidemment pas ces problèmes là.

5.1.2 Lenteur

Sur les ordinateurs de l'UFR, nous avons constaté certaines lenteurs avec notre programme.

Sur un ordinateur doté d'une carte graphique Nvidia Geforce GTX 950M, nous n'avons pas vraiment de soucis de lenteur en revanche.

Il se peut qu'il y ait des optimisations à faire pour éviter des gros calculs au GPU.

5.1.3 Brouillard exponentiel

Nous avons essayé d'implémenter un brouillard exponentiel qui dépend de la position de la caméra et de la profondeur mais nous n'obtenions pas un résultat convenable.

Nous nous sommes donc rabattu sur un brouillard linéaire.

5.2 Améliorations

5.2.1 Concernant le brouillard

Il aurait pu être intéressant de créer un brouillard selon la hauteur des sommets (et pas seulement selon la profondeur par rapport à la vue), se rapprochant ainsi des phénomènes de brouillard que l'on constate naturellement sur les hauts sommets.

5.2.2 Créer des reliefs locaux

Une autre idée (mais qui aurait pris du temps) aurait été de faire du displacement mapping (au mieux) sur les zones à éventuel relief local comme la roche.

5.2.3 Paramètres du bruit

Nous avions pensé à modifier les paramètres du bruit individuellement selon une constante dépendant du vecteur de mouvement, malheureusement nous ne l'avons pas réalisé.