BSI TP2: Application de textures

Loïc Simon

August 20, 2015

L'objectif de ce TP est de mettre en place les techniques basiques de placage de texture, à savoir

- le chargement d'une texture dans le GPU (à partir d'un fichier TGA),
- les directives de coloration dans le fragment shader
- la définition de coordonnées UV (dans le VAO),
- le filtrage (magnification et minification),
- le mipmapping

1 Exercices

NB: Pour l'exercice 1, vous devrez modifier le fichier `utils/textures.cpp`, et pour les suivant le fichier `Textures/textures.cpp`. Il s'agit bien de deux fichiers différents. Vous aurez aussi à modifier le fragment shader `texture.f.glsl`.

1.1 Exercice 1: Chargement d'une texture

Dans le fichier `utils/textures.cpp` remplir le code de la fonction `loadTGATexture` pour effectuer le chargement d'un fichier de texture dans le CPU puis la stocker dans le GPU. Vous utiliserez notamment les routines `read_tga` pour charger la texture dans la mémoire CPU puis `glTexImage2D`, pour la stocker dans le GPU.

Utiliser la routine `glTexParameteri` pour contrôler les paramètres de wrapping (extension de la texture en dehors de son domaine naturel $[0, 1]$) et de filtrage.

Dans cet exercice, vous aurez aussi besoin d'éditer le fichier `texture.f.glsl` pour mettre en place la coloration par texture dans le fragment shader. Vous devrez utiliser la routine GLSL `texture`.

Lorsque ces deux étapes seront réalisées, la scène sera uniformément texturée par un damier. Vous devrez alors éditer le fichier `Textures/textures.cpp` dans la fonction `draw()` pour associer une texture de dé a certains objets de la scène. Pour cela vous devrez associer la bonne texture a l'objet uniform de te type sampler2D du programme de shader utilisé pour texturer.

Q1. Expliquez succinctement et avec vos propres mots le rôles des paramètres de la fonction `glTexImage2D`.

Q2. Essayez plusieurs paramètres de wrapping (extrapolation) et de filtrage (interpolation) différents, et expliquez leurs effets. Illustrez par des captures d'écran.

1.2 Exercice 2: Coordonnées uv

Ouvrez le fichier `dice_texture.tga`, et observez le. Remplacez les valeurs du buffer `correct_uv_buffer_data` pour que chacun des numéros se plaque correctement sur une face du cube. Vous pourrez aussi télécharger d'autres textures qui vous plaisent et les plaquer sur le cube. Pour cela vous les convertirez en `tga` en utilisant la commande suivante:

```
convert input.jpg -type TrueColor output.tga
```

Q3. Commentez sur la difficulté pour obtenir un mapping cohérent pour différentes textures.

1.3 Exercice 3 et 4: Mip-mapping et filtrage anisotrope

Dans la fonction `drawPlane`, gênerez des mipmaps et activez les. Ensuite, mettez en place un filtrage anisotrope.

Q4. Expliquez la différence entre la magnification et la minification (dans le contexte du placage de textures).

Q5. Expliquez quels défauts visuels le mipmapping cherche a éliminer. Expliquez le principe de fonctionnement de cette technique, et notamment pourquoi elle permet d'éliminer les effets indésirables susmentionnés.

Q6. De même, expliquez le rôle et le principe du filtrage anisotrope.

1.4 Exercice 5: Fragment shader

Changez la définition de fragColor, pour implementer des rendus originaux (par exemple animés). Vous pourrez utiliser la texture, la couleur, et toutes autres entrées (`in` ou `uniform`) que vous aurez passées au shader.

Q7. Décrivez ce que vous avez entrepris ici.