OpenGL, Utilisation du GPU

CPE – David Odin

1 Quelques liens utiles

— Le site officiel concernant OpenGL : http://opengl.org/.
— La documentation en ligne des fonctions OpenGL 2.1 : http://www.opengl.org/sdk/docs/man2/.
— La documentation en ligne des fonctions OpenGL 3.3 : http://www.opengl.org/sdk/docs/man3/.
— Un résumé des fonctions OpenGL et GLSL (pour OpenGL 3.2 et GLSL 1.5) : http://www.khronos.org/files/opengl-quick-reference-card.pdf.

2 Traitement d'image simple

Le programme `image_simple` propose une implémentation d'un filtre de changement de contraste.

Pour cela, il charge une image dans la texture courante, puis dessine un *quad* de la même taille que la texture, avec un *fragment shader* adapté pour filtrer à la volée.

Travail à effectuer : *Analysez le programme en question (le source en C++ et le fragment shader) et assurez vous de bien le comprendre.*

Question 1 : *Modifiez le fragment shader pour en faire des filtres différents : noir et blanc, sépia, saturation, etc. Vous rendrez au moins 2 fragment shaders pour cette question.*

3 Traitement d'image avec convolution

En modifiant un peu le *fragment shader*, on peut en faire un filtre convolutif.

Question 2 : *Modifiez le fragment shader pour en faire un filtre de flou comme présenté dans le cours. Rien à rendre pour cette question, sauf si vous proposez un autre type de filtre que celui donné en cours.*

Question 3 : *De la même façon, implémentez un filtre de détection de bord, ou un autre filtre convolutif. Rendre au moins un fragment shader et une capture d'écran.*

4 Utilisation des FBO

Dans cette partie, nous allons utiliser un FBO afin de créer un rendu en deux passes. La première passe dessinera un cube en rotation, texturé avec une image dans un FBO. La deuxième passe utilisera la texture associée au FBO pour la plaquer sur un autre cube que l'on dessinera dans la fenêtre.

Pour cela, vous complèterez le fichier `main.cc` de l'archive FBO.tar.gz fournie, en utilisant notamment les fonctions suivantes :

— glGenTextures(...);
— glBindTexture(...);
— glGenFramebuffers(...);
— glBindFramebuffer(...);
— glFramebufferTexture2D(...);

Regardez bien les documentations de ces fonctions, et suivez les commentaires présents afin de compléter le code (il faut

remplacer les commentaires commençant par `// ***` par le code correpondant).

Assurez-vous également de bien comprendre les fonctions suivantes (non présentées en cours) :

— glGenRenderbuffers(...);
— glBindRenderbuffer(...);
— glRenderbufferStorage(...);
— glFramebufferRenderbuffer(...);

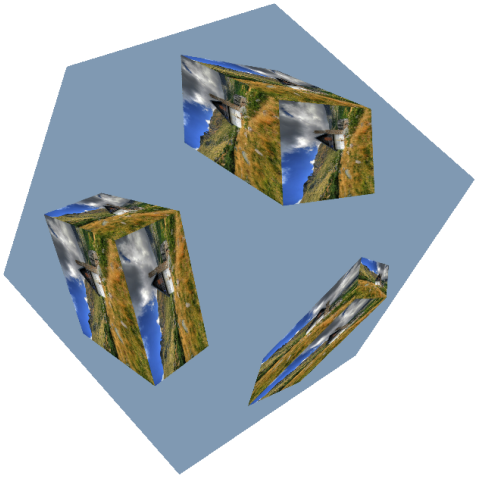

FIGURE 1 – FBO : un cube dans un cube.

Question 4 : *Essayez d'obtenir une figure semblable à la figure 1, vous pouvez bien entendu ajouter votre touche personnelle en changeant la texture, la forme de ce qui est dessiné dans le fbo ou sur la fenêtre. Dans tous les cas, vous rendrez votre code et une capture d'écran.*

5 Moteur de particules avec un transform-feedback

Dans cette partie, nous allons implémenter un moteur de particules et l'afficher, en utilisant la fonctionnalité "transform-feedback" apparue dans OpenGL 3.x.

Chaque particule a une position et une vitesse propre.

L'exemple donné dans `transform-feedback.tar.gz` ne manipule que les positions.

Il vous faudra dans un premier temps comprendre le comportement du programme donné, ainsi que le code présent dans le fichier shader `data/tf.vert`.

Question 5 : *Ajoutez la gestion des vitesses, par exemple en ajoutant une accélération due à la pesanteur, et rendez le tout plus joli !*

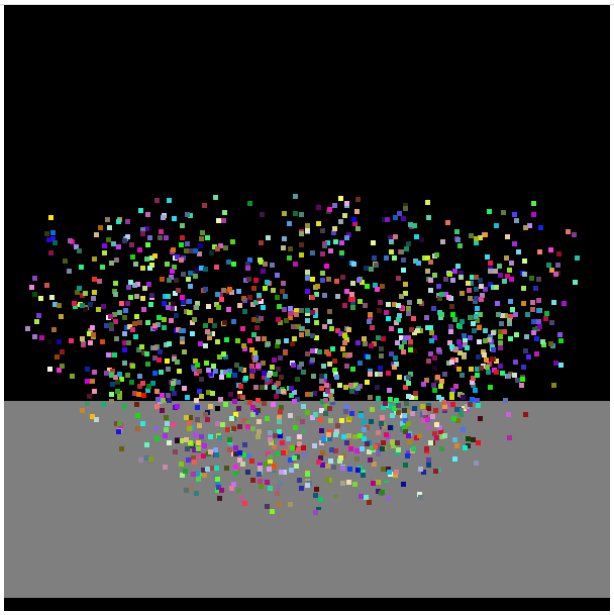

FIGURE 2 – Quelques particules.