GPGPU
geometry shaders
CPE

5ETI IMI

1 But

L'objectif de ce TP est d'apprendre à utiliser les geometry shaders.

2 Prise en main de l'environnement

2.1 Compilation

Question 1 *Compilez le code, assurez vous de voir un maillage gris.*

Question 2 *Dans* glhelper.h*, qui reprend le travail effectué en TP1, assurez vous de comprendre le role de chaque fonction.*

3 Gestion de plusieurs programmes

Question 3 *Créez un second programme contenant un fragment shader différent, et à l'aide d'une touche clavier changez la coloration du maillage.*

4 Gestion des uniformes

Question 4 *Regardez le passage de paramètres uniformes (paramètre commun à tous le programme GPU) pour la camera. Sur le même modèle, modifiez votre fragment shader pour qu'il évolue dans le temps. Vous pouvez utiliser la librairie* chrono *de la stl.*

```
auto t_start = std::chrono::high_resolution_clock::now();
...
auto t_now = std::chrono::high_resolution_clock::now();
float time = std::chrono::duration_cast<std::chrono::duration<float>>(t_now - t_start).count();
```

5 Ajout des geometry shaders

Question 5 *À la manière de* create_program_from_file(. . .)*, ajoutez une fonction pour créer un programme avec en plus, un geometry shader.*

Question 6 *Modifiez le programme principal afin de prendre en compte le geometry shader* basic.gs*. Compilez, lancez le programme, qu'obtenez vous, est-ce prévisible?*

Question 7 *Créez un geometry shader pour montrer une vue éclaté de l'objet –il suffit de déplacer les sommets dans le sens de la normale.*

Question 8 *Créez un geometry shader afin de visualiser les normales sous la forme de ligne par dessus le maillage. Il vous faudra utiliser deux programmes dans la fonction d'affichage, l'un pour le maillage, l'autre pour les normales.*

Question 9 *À partir d'un geometry shader, calculez la normale à la surface et affichez la couleur associé dans fragment shader (passage de paramètre en shaders).*