Antoine Moineau 3F1

TP 5 - 6 : introduction à la segmentation

Ce TP a pour but d'approfondir la manipulation de la librairie CGAL en construisant un outil de déformation continue de maillages.

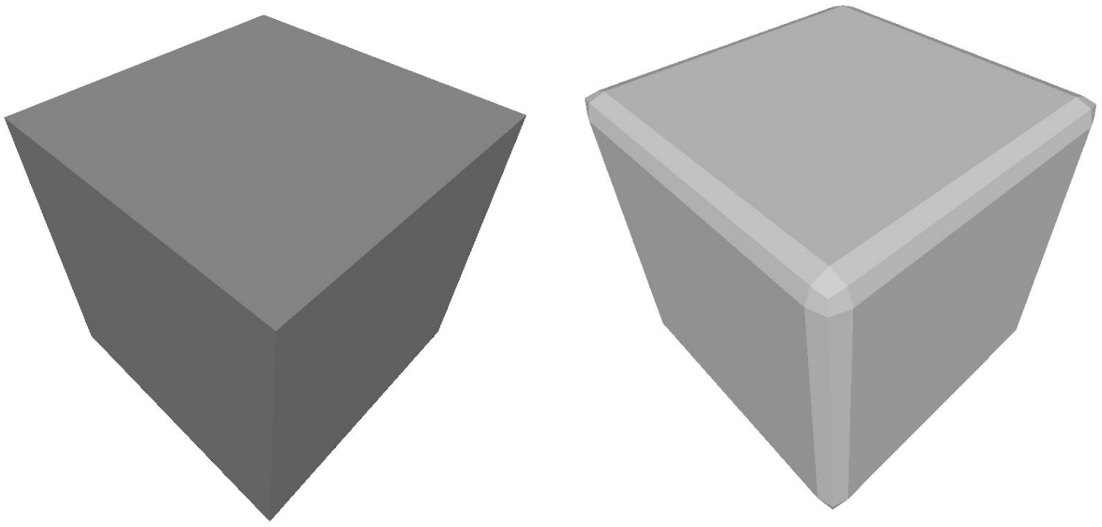
Mise en place

Le programme est composé d'un seul fichier CPP qui prend en entré un maillage au format .OFF et qui crée une version lissée de ce maillage, **sans le modifier**. Pour ce faire nous allons travailler sur des *maps* qui associeront un point du maillage à une coordonnée. La première étape consiste donc à récupérer toutes les coordonnées initiales pour créer la *map* que l'on utilisera comme point de départ. De cette façon, chaque méthode de lissage prend en entrée, au moins, le maillage et un jeu de coordonnées puis retourne un jeu de coordonnées lissées. Cela évite de modifier le maillage ou d'en faire des copies et permet d'interchanger les méthodes facilement.

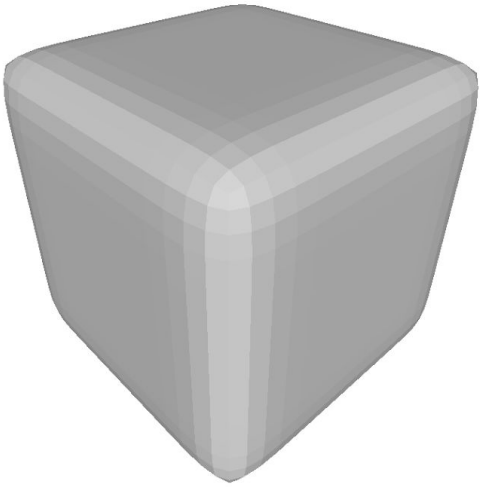
Lissage Laplacien

Le premier lissage mis en place est un lissage Laplacien simple. Le lissage Laplacien est une procédure qui parcourt l'ensemble des sommets du maillage, et calcule pour chacun d'eux une nouvelle position à partir de la moyenne des positions des sommets voisins. Cela se fait facilement avec deux parcours imbriqués, le premier pour visiter chaque point du maillage et le second pour visiter tous les voisins et faire la somme de leurs coordonnées. On divise ensuite par le nombre de voisin et on ajoute ce nouveau point dans le jeu de coordonnées à retourner.

On obtient le résultat suivant (droite avant lissage, gauche après):



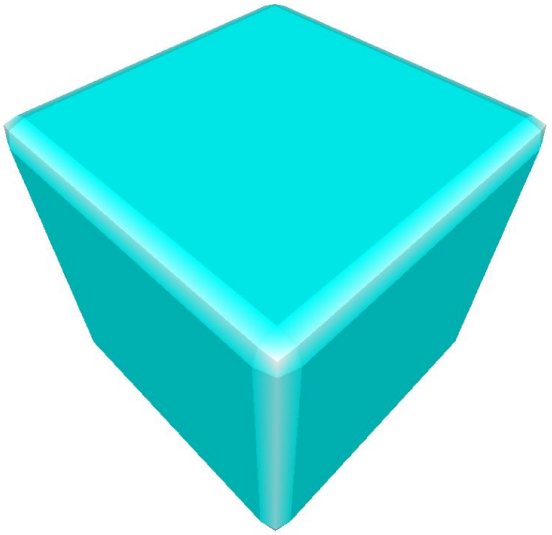
Ce lissage marche bien mais reste très limité étant donné qu'il ne prend en compte que les voisins directes. On peut éventuellement l'appliquer de manière successive pour obtenir un lissage un peu plus intéressant. Voilà ce que l'on obtient après 5 itérations:



Visualisation

Comme pour le projet précédent il faut exporter le résultat en .OFF afin de pouvoir afficher le maillage comme dans les images ci-dessus. La seule différence est qu'il faut utiliser les coordonnées de la *map* issue du lissage qui nous intéresse.

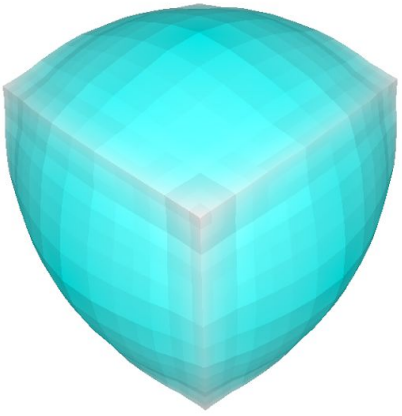
Cependant, afin de mieux visualiser les déformations, nous souhaiterions colorer le maillage en fonction de l'importance du déplacement local. Pour ce faire, nous n'allons pas définir la couleur par face mais par sommet en utilisant le format COFF. Dans les méthodes de lissage, lors du calcul de la nouvelle coordonnée, on mesure la distance entre cette coordonnée et la coordonnée précédente et on conserve le déplacement maximum et minimum. Ensuite, dans la méthode de sauvegarde, on peut utiliser le rapport entre le déplacement d'un point et les extremums pour faire varier la couleur. Cela nous donne le résultat suivant :

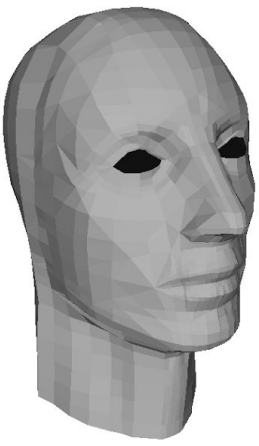


Rayon du voisinage

Maintenant nous aimerions créer un lissage plus intéressant et sur lequel on aurait plus de contrôle. Au lieu de faire un lissage Laplacien sur les voisins directs, nous allons prendre en compte tous les voisins dans une boule euclidienne.

Pour cela nous allons réaliser, à partir de chaque sommet, un parcours en profondeur à la surface du maillage qui s'arrêtera quand il sortira de la boule. Cela nous donne le résultat suivant:

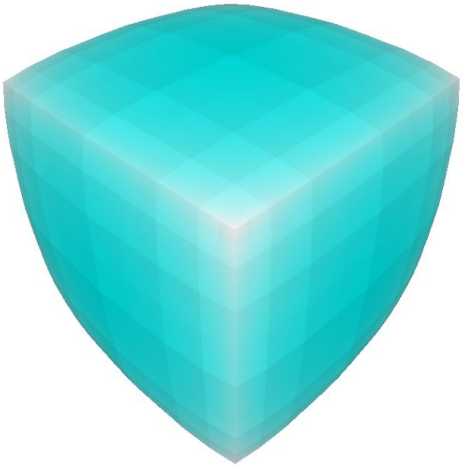




On remarque que ce lissage paraît plus intéressant sur certains modèles mais peut donner des résultats incohérents sur des arêtes vives comme comme dans le cas du cube. De plus, un rayon trop grand donne des résultats aberrants.

Rayon du voisinage

Afin d'essayer de résoudre les problèmes évoqués précédemment, nous allons ajouter une notion de pondération. Plus un sommet est loin du point à modifier moin il aura d'influence sur la déformation :



On constate que cela fonctionne et que les résultats sont bien meilleurs dans les deux cas.

Conclusion

Sur la totalité des tests que nous avons réalisé le dernier filtre offre des résultats bien plus intéressants, notamment car on peut jouer sur 3 facteurs pour obtenir le résultat désiré: le rayon du voisinage, la pondération et les itérations successives.