Antoine Moineau 3F1

TP 4 : introduction à la segmentation

Ce TP a pour but d'apprendre à manipuler la librairie CGAL en pratiquant divers segmentations sur un maillage.

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 colorée de ce maillage en fonction des segmentations, **sans le modifier**. L'objectif étant d'expérimenter, il faut que le code soit modulaire afin que les méthodes de seuillage et de calcul soit facilement interchangeables.

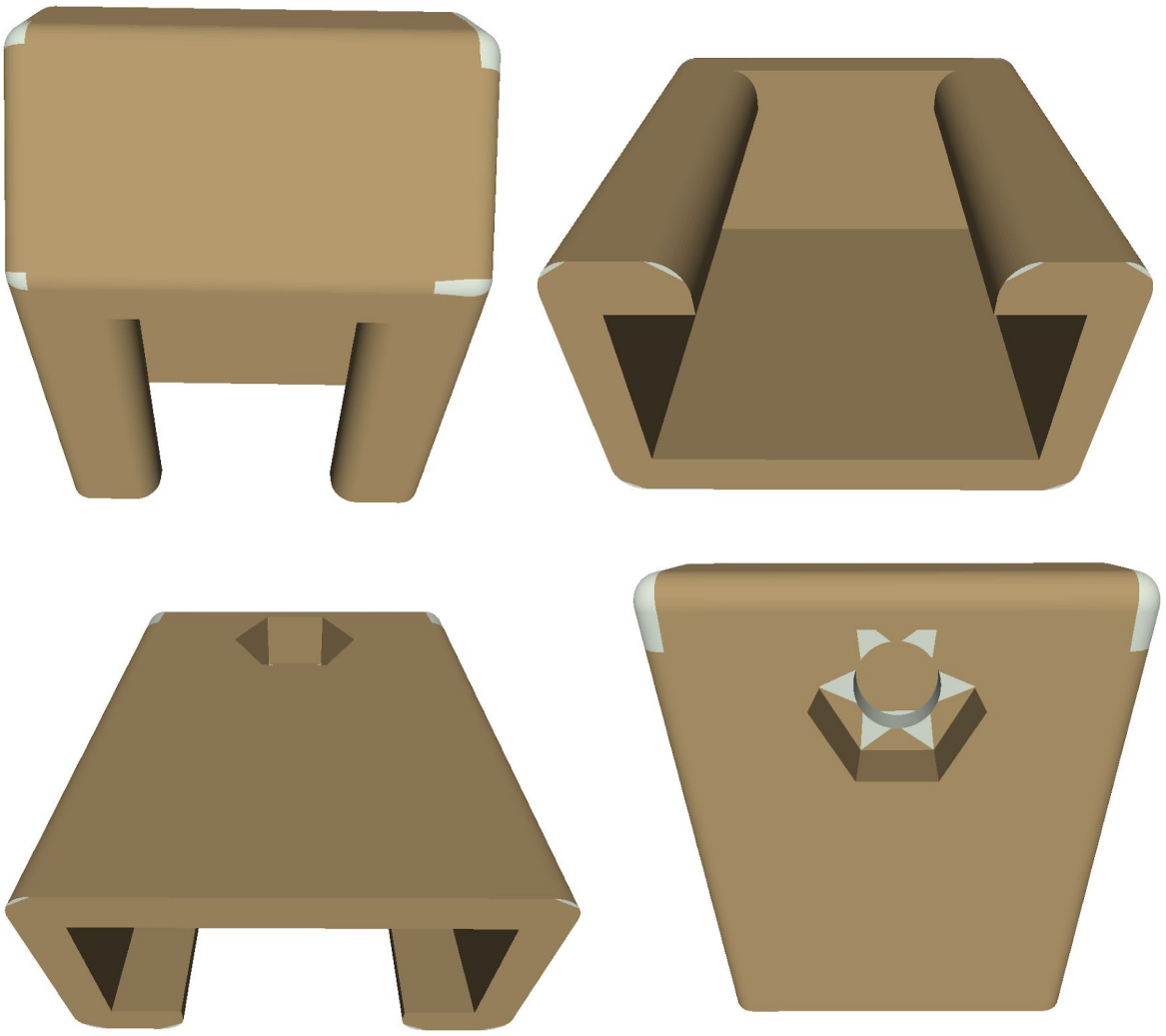
La première méthode implémentée consiste donc à créer un fichier .OFF à partir d'un objet *CGAL::Polyhedron_3<Kernel>* et d'un plan déterminant la segmentation de ce maillage. Ce plan prend la forme d'une *std::map <Polyhedron::Facet_handle, int>* où chaque face est associée à une catégorie. La méthode attribue des couleurs aléatoires à chaque catégorie ce qui permet de facilement les distinguer.

Introduction au seuillage

Nous avons commencé par une segmentation selon un seuil très simple : le périmètre. Pour faciliter la visualisation nous avons décidé que les faces seraient divisés en seulement deux catégories, celles dont le périmètre est supérieur à la moyenne et les autres.

Pour ce faire il faut une première méthode qui calcul le périmètre de chaque face ainsi que la moyenne et une seconde méthode qui attribue les catégories aux faces. Ces deux méthodes sont conçues afin de pouvoir facilement changer l'une des deux pour avoir des seuils différents. Par exemple, on peut remplacer le calcul du périmètre par celui de l'air sans avoir à modifier la méthode de seuillage.

On obtient le résultat suivant:



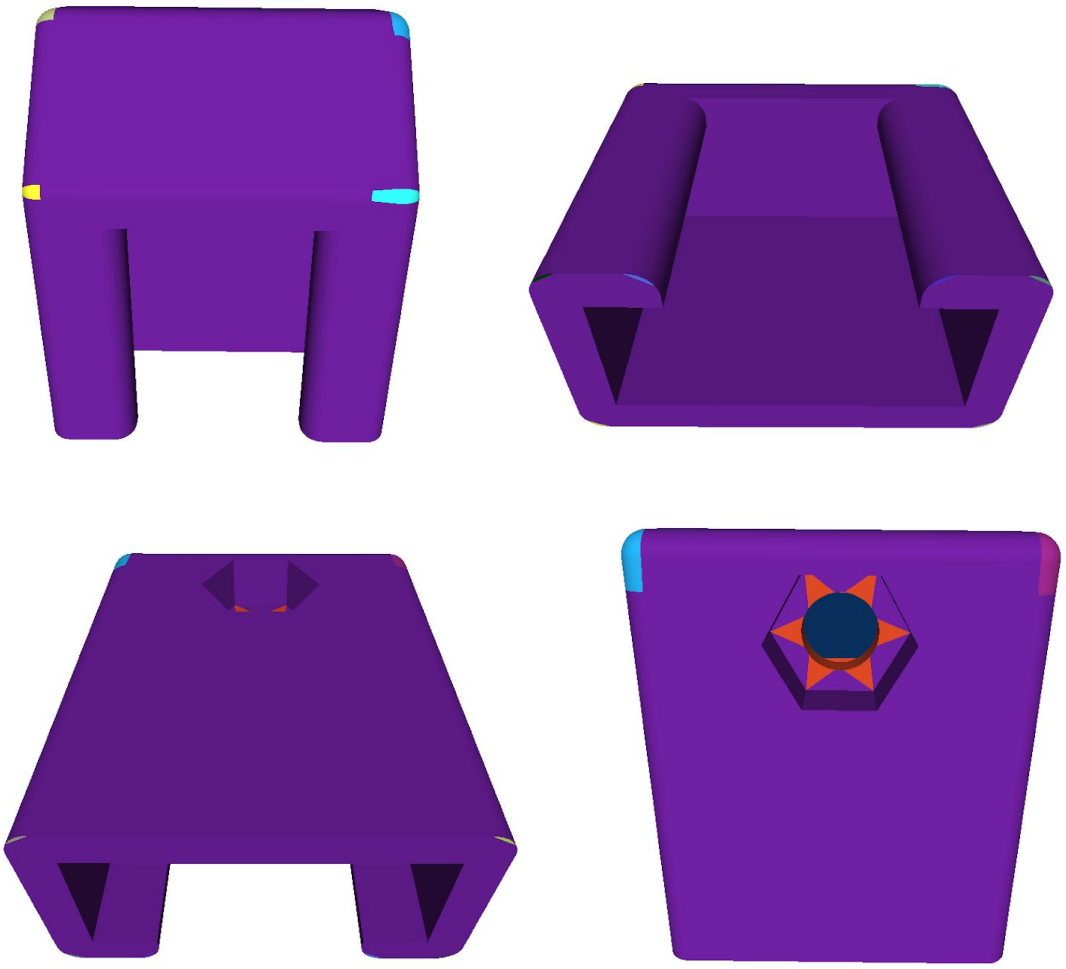
Intégration des composantes connexes

Afin de complexifier un peu le seuillage et de le rendre plus intéressant nous avons décidé de créer une méthode qui va prendre en entrée le maillage et un plan de segmentation pour créer un nouveau plan, **sans modifier l'ancien**. Encore une fois, la modularité est importante et il faut pouvoir remplacer ou supprimer cette étape sans modifier le reste du traitement.

Nous avons décidé que la sous-division des classes se fera en suivant les composantes connexes définies par les groupes issus de la segmentation précédente. Pour ce faire:

- On part d'une face, on l'empile et on la marque comme visité.
- On dépile les faces jusqu'à ce que la pile soit vide.
- A chaque fois que l'on dépile une face on l'attribue à la nouvelle catégorie et on empile toutes ses faces voisines, non visitées, de la même catégorie et on les marques comme visitées.
- Une fois la pile vide on a finit une sous-division et on peut répartir d'ailleur.

Une fois cette étape terminée il suffit de donner ce nouveau plan à la fonction de création pour obtenir le résultat suivant:



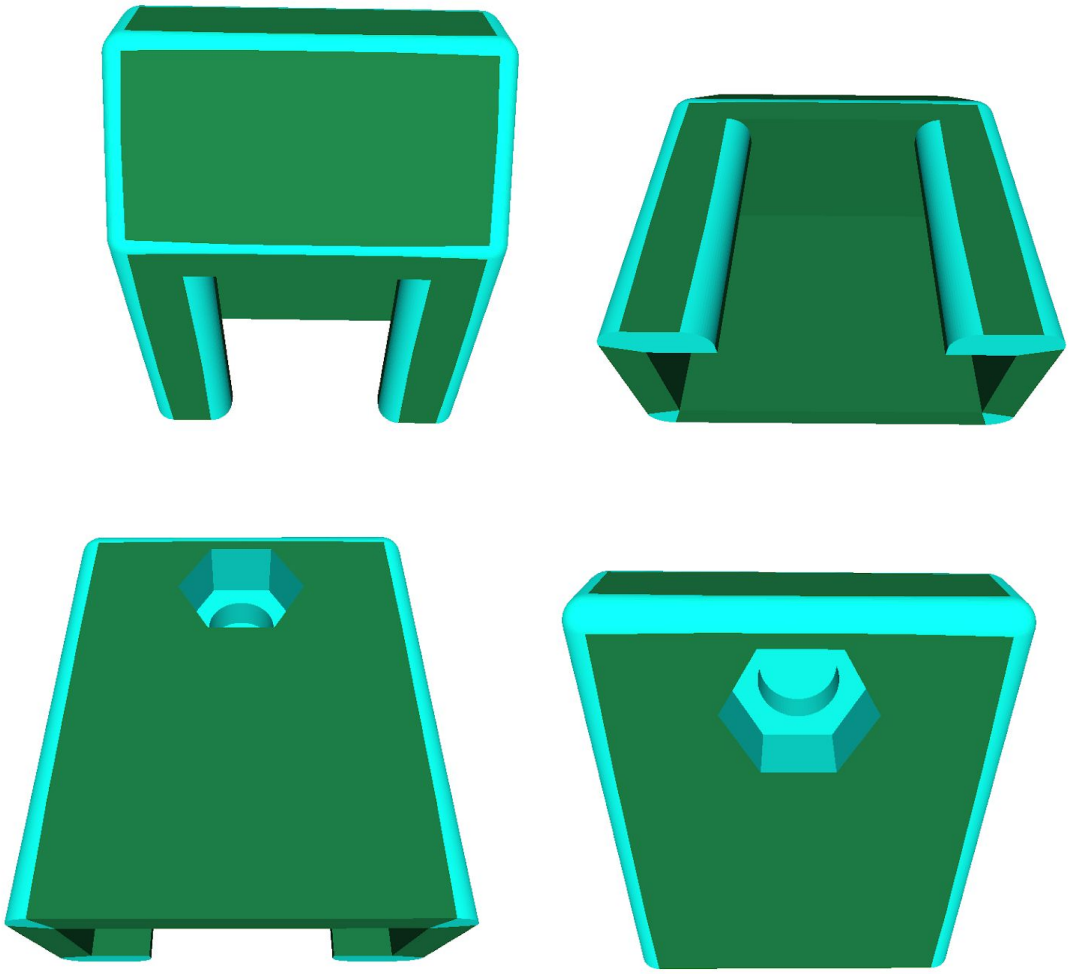
On remarque que les zones qui étaient précédemment de la même couleur sans se toucher sont maintenant de couleur différentes.

Choix du seuil

Maintenant que nous avons une chaîne de traitement complète, nous pouvons intégrer d'autres types de seuillage. Une méthode plus complexe et fréquemment utilisé est la méthode d'Otsu.

Dans cette méthode, on commence par calculer un histogramme de la valeur à seuiller, puis on calcule le seuil optimal à l'aide de la variance des deux classes. Afin d'adapter cette technique au cas des maillages, il est important de calculer un histogramme pondéré, où chaque triangle « pèse » en fonction de son aire dans l'histogramme. Ici, nous avons choisi de donner plus d'importance à certaines faces en fonction de leur aire.

Voilà le résultat lorsqu'on applique cette méthode au même maillage. La mesure d'origine reste celle du périmètre.



On remarque que la segmentation paraît nettement plus intéressante de notre point de vue. Si l'on applique la segmentation connexe par dessus, la zone verte est divisée en trois couleurs et la zone bleu en deux.

Conclusion

Sur l'exemple utilisé ici il est clair que le seuillage par la méthode d'Otsu et bien plus intéressant pour isoler des parties de manière sémantique. Cependant, il n'existe pas une méthode universellement meilleur, sur d'autre maillage la première méthode donne en réalité des résultats plus intéressants. Cela dépend de la structure du maillage et c'est pour cela qu'un programme modulable est important afin de pouvoir tester différentes méthodes facilement.