


TP4 – Segmentation de maillages

Rémi Alègre – ISIMA 3^{ème} année F1, idée originale de Jean-Marie Favreau

 **Ce TP est évalué, en binôme ou monôme.** Je vous demande d'envoyer à l'adresse remi.alegre@ext.uca.fr les livrables suivants...

- Archive contenant les différents programmes réalisés (code source, cmake, exécutables...)
- Un court rapport (de l'ordre de 4 à 5 pages) décrivant vos expérimentations, illustré de captures d'écran de maillages colorés ainsi qu'une petite partie discussion de vos résultats

au plus tard le  **mardi 21 février 23:59** !

 **Une utilisation adéquate de git et GitLab/GitHub est grandement recommandée pour ce TP et les suivants.**

1 Prérequis vis-à-vis du TP précédent

Pour mener à bien correctement ce TP, il est préférable d'avoir déjà développé

- au moins 3 ou 4 fonctions de mesures sur des maillages, par exemple :
 - périmètre d'une face
 - angle entre la normale d'une face et l'axe des X, Y, ou Z
 - aire de chaque face
 - moyenne des angles formés par la normale de la face courante et chacune des normales de ses faces voisines.
- une fonction pour écrire un fichier .off coloré en fonction d'une mesure calculée sur chaque face


Bien sûr, un peu de temps pourra être utilisé pendant la séance ou en travail personnel sur ces points-là, mais la partie évaluée portera avant tout sur les points suivants, donc attention à ne pas se laisser emporter !

2 Introduction au seuillage

Lors de la séance précédente, nous avons calculé différentes mesures sur les faces, telle que leur périmètre.

La visualisation par dégradés de couleur n'est pas toujours simple à lire et ne permet pas directement d'apporter du sens à notre maillage.

Nous allons donc explorer la piste du seuillage : séparer les faces en différentes classes suivant que leur valeur est plus grande ou plus petite que le seuil choisi.

 **Travail à réaliser** : Écrire une fonction qui attribue pour chaque face un entier suivant qu'il est dans l'une ou l'autre des deux classes de cette segmentation, en ayant fixé le seuil par une méthode simple (par exemple on pourra utiliser la moyenne des mesures calculées par les faces, ou expérimenter avec la médiane).


 Signature proposée :

```
Facet_int_map simpleThreshold(Polyhedron & mesh, Facet_double_map & values)
```

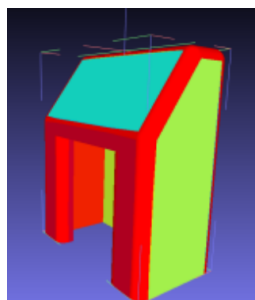
où on aura défini...

- `typedef std::map<Polyhedron::Facet_handle, double> Facet_double_map;`
- `typedef std::map<Polyhedron::Facet_handle, int> Facet_int_map;`

3 Génération de maillages colorés

 **Travail à réaliser** : Adapter votre travail du TP précédent pour exporter un fichier .OFF où chaque face du maillage se voit affectée une couleur différente en fonction de sa classe de segmentation. (On supposera qu'on n'est pas limité à seulement 2 classes de segmentations).

On pourrait par exemple définir un tableau de N couleurs distinctes, pour que chaque indice de classe puisse aller "piocher" sa couleur dans le tableau.

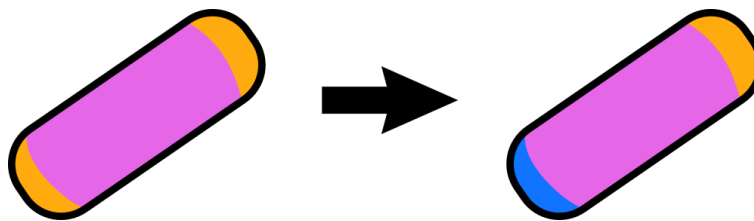


4 Intégration des composantes connexes

En observant le résultat de cette segmentation sur différents maillages, on observe qu'à la surface de l'objet, plusieurs composantes connexes sont colorées de la même

couleur. On se propose donc, étant donné une segmentation, de produire une nouvelle segmentation qui respecte l'originale, tout en attribuant à chaque composante connexe originale un identifiant différent.

Exemple : si un objet de forme allongée est composé de grandes faces au centre, et de petites faces aux deux extrémités, une première segmentation par seuillage à partir du périmètre aura produit deux classes de faces (les petites et les grandes). Une fois introduite la notion de composante connexe, on aura alors trois classes : une classe par extrémité, et une classe centrale.



Travail à réaliser : écrire une fonction qui étant donné un maillage et une segmentation produit une nouvelle segmentation qui sépare en différentes classes les composantes connexes de la segmentation initiale.

Signature proposée :

```
Facet_int_map segmentationParCC(Polyhedron & mesh, const Facet_int_map & segmentation)
```

Indications : on va parcourir le maillage, en regardant successivement toutes les facettes. Pour chaque facette, si elle n'a pas été vue, on la marque comme vue, puis on parcourt les facettes voisines, avec un appel récursif si elles sont dans la même classe de segmentation que la première facette. On peut utiliser pour cela un parcours des demies-arêtes autour de la face, et pour chacune de ces demie-arêtes, on utilise `opposite()` pour obtenir la demie-arête opposée, puis `facet()` pour obtenir la facette adjacente.

5 Expérimentation

On se propose d'expérimenter diverses techniques de segmentation, divers paramètres sur divers objets, et de les illustrer avec quelques exemples (maillages colorés à inclure dans le rapport).

On s'intéressera également à discuter des différentes manières de choisir le seuil (manuel, moyenne, médiane...) et l'intérêt de distinguer les composantes connexes ou non.

Quelques sujets d'expérimentation et de discussion..

- Quel choix de mesure locale est plus intéressant ? Est-ce que cela dépend du type de maillage traité ?
- Comment définir des seuils pertinents ?
- Comment associer plusieurs mesures ou plusieurs segmentations pour obtenir des segmentations plus complexes, est-ce intéressant pour détecter des caractéristiques plus complexes sur nos maillages ?
- Si l'approche locale a été étudiée (question 6), quels sont ses avantages et inconvénients ?

Bonus : il serait intéressant d'orienter ces expérimentations autour d'un cas d'usage "concret".

Quelques idées de cas concrets...

- Supposons un maillage de mobilier (table, chaise, cube...) ou de terrain, nous cherchons à déterminer les zones sur lesquelles il est possible de déposer un objet sans qu'il tombe (ou d'y planter un arbre ou un bâtiment...). Autrement dit, on souhaite mettre en évidence les parties du maillage plutôt orientées vers le haut, et plutôt planes.
- Supposons que l'on considère qu'un maillage est de qualité si les triangles qui le composent ont des angles équilibrés, et des aires relativement semblables. Identifier les parties de qualité et de mauvaise qualité dans un maillage.
- Il serait intéressant de pouvoir identifier les faces équidistantes à un point (distance euclidienne, ou mieux... [géodésique](#) !) par exemple pour déterminer quelle partie d'un donut sera mangée en premier, puis en dernier si on commence à le déguster en partant d'un point donné.

Vous pouvez aussi bien entendu proposer vos propres cas d'usage.

6 Question bonus : Approche locale

💀 Cette question est un peu plus avancée. Assurez-vous d'avoir toutes les clés en main pour boucler les questions précédentes avant de commencer à traiter celle-ci ! Bien sûr, tout travail sur cette question même expérimental sera dûment récompensé !

Plutôt que d'avoir recours à un seuillage global sur tout le maillage, on peut essayer de parcourir le maillage de faces adjacentes en faces adjacentes (comme pour la question

précédente, en parcourant les demi-arêtes, puis en accédant à la facette adjacente via la routine "`->opposite()`"->`facet()`"), en affectant un même identifiant de classe à tous les voisins dont la différence de périmètre est inférieur à un seuil T , si la différence est supérieur, on ignore la face. Si on continue à parcourir les faces dont une classe n'a pas été encore affectée par le premier parcours, jusqu'à avoir parcouru l'intégralité du maillage, on obtiendra une segmentation plus complexe.

Vous n'avez rien compris ? N'hésitez pas à interpellier votre professeur pour plus d'information !