AM-MIRI: Lab 2 Exercises

Here is a list of exercises for the second session of the lab. They should all be solved as separate modules (to be imported to blender). You may use the `Mesh` class or the `bmesh` class; your choice, but be aware of what you are using. If the result requested is provided directly by the blender API objects that you are using (for example the area of a face) then implement your own (and compare, if you happen to know that blender has it too). You may assume that your code is executed with exactly one object selected. Use `info_mesh.py` as a starting point, or as example of simple things one may do with a `Mesh`. You may want to browse the templates in the blender editor as well.

The file `Project_1_dataset.zip` provides some sample objects on which to test your functions.

Also, **please round off to 3 decimal places** all floating-point results. For example, you may define

```
def r(a):
    return int(a*1000+0.5)/1000.0
```

and print results rounded by this function.

Each function should additionally report the time consumed by its execution (as done for `processa_malla(mesh)` in `info_mesh.py`). Naturally, you are expected to produce efficient code; if something requires an $O(n)$ effort, a solution involving $O(n^2)$ operations is not adequate...

Turn in your solutions to exercises 7 and 9 by **March 24$^{\text{th}}$, 2019**, via an ad-hoc "Pràctica" that you will find in the racó

Exercise1 — Centroid

To warm-up, write a function that computes (and prints) the centroid of the vertices of the selected object. In these exercises, you may find advisable to write a function that returns the desired result, plus a "driver" function (perhaps the `main` of a module, for example) that invokes the function and prints the result. In this way, you may reuse your code more easily in other exercises.

Exercise2 — Analysis

Analyze the result of your solution to Ex. 1 when applied to the model in `hollow-icosas.blend`. Given the symmetry, you may expect the barycenter to sit square at the origin. What happens? Think of the data structures in blender (how an object is stored...).

Exercise3 — Vertex valences

Write a function that prints out the minimum, maximum and average valence of all vertices in the selected mesh. Beware inefficient code! Should return an answer quickly for all exmples supplied.

Exercise4 — Manifold and non-manifold edges

Write a function that computes and prints the number of boundary, manifold and non-manifold edges in the selected model. Notice that you will need to build some auxiliary data structure to determine neighboring polygons if you want your code to be efficient (and yes, you want that).

Exercise5 — Convex and concave edges

Write a function that computes the number of convex and concave edges of the selected object. Notice that you may benefit here from code written for Ex. 4

Exercise6 — shells

Write a function that prints the number of shells of the selected object. Try to write efficient code. (Hint: does union-find ring a bell?)

Exercise7 — genus

Using your result in Ex. 6 and the fact that Blender's model does not include rings, use the euler formula to compute the genus of the selected object.

Exercise8 — Surface area

For the selected object, compute the surface area of each face, an print a line for each face containing your computed area and the area stored in the `area` attribute of the polygon.

Exercise9 — Total volume

Write a function to compute (and to print) the total volume of the selected object. Different connected components should be added, but shells that do not form a 2-manifold without boundary may be ignored (what would be their volume...)

Exercise10 — Center of mass

Compute the center of mass for a given object (we assume a 2-manifold boundary and a homogeneous interior).