Direct visibility of Delaunay triangulation elements

MTB Challenge - Winter Term 2018/19

Michal Mašek, Vít Losenický

October 29, 2018

1 Motivation

Working with numerical simulators is nearly impossible with a smooth shape of objects of interests. These objects have to be discretized into a finite set of nodes which approximately corresponds to an original body. The number of nodes and their size determine precision of details. In order to model an arbitrarily shaped object appropriately the Delaunay triangulation is often chosen to generate triangular mesh grid [2].

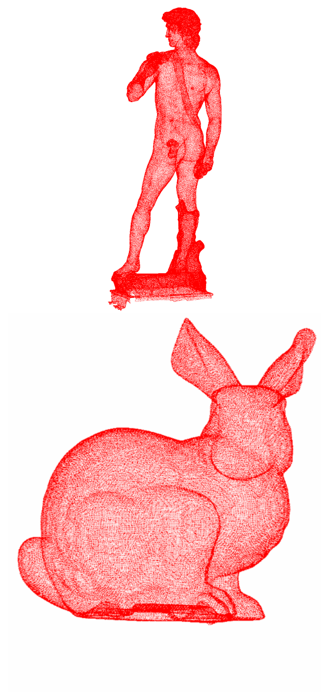
The problem to deal with in this challenge appears with visualization of the mesh grid, when it is necessary to analyze which nodes (or triangles) are visible and which are hidden behind those placed in foreground. The left panel of Figure 1 shows all points of object, where is not possible to decide of the objects are looking forwards or backwards. The figures were reprinted from [1], where a resolution to this remedy is proposed introducing an operator distinguishing directly visible and invisible points. The solution is shown in the right panel of Figure 1. Analogical procedure for a triangular mesh is the core of the challenge.

2 Competition Setting

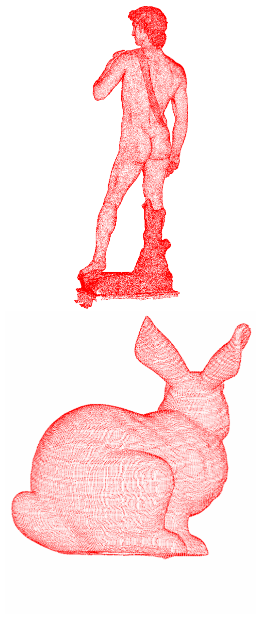
The challenge has several subtasks:

2.1 Task 1: Ordering of triangles for direct visibility

The first task is to determinate triangles' order of visibility from predetermined point of view. Proper ordering is crucial for plotting, by example in LaTeX and Ti*k*Z, when the triangles should be drawn from the most distant one to the closest one. An example of the correct and incorrect ordering is shown in Figure 2.



(a) Complete set



(b) Set of visible points

Figure 1: Object discretized into points [1].

2.2 Task 2: Direct visibility of the mesh

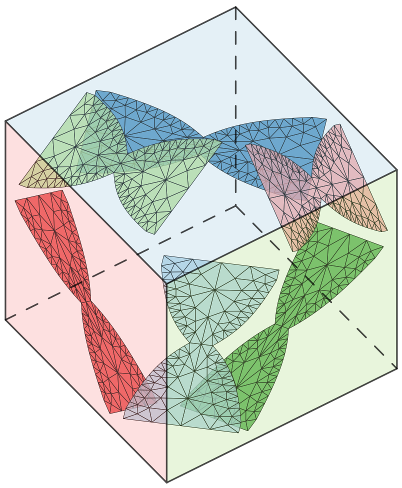
In case of opaque figure it is not necessary to plot triangles which are not visible. The second task is to designate visible, non-visible, and partly visible triangles and to draw only the visible part of the structure. The required procedure should be similar to `cleanfigure` in Matlab2TikZ package which is capable to remove all unused primitives and simplify thus the figure.

2.3 Task 3: Orthogonal projection

For a given camera position compute an orthogonal projection of object to a plane - see an illustrative Figure 3. Calculate area of the projection.

2.4 Bonus Task: Shadowing

Let us imagine a point source of light located at a given azimuth and elevation. The task is to compute an incidental shadow on each triangle expressed, for example, to be assigned by different colors from grey-scale colormap to each triangle.


(a) Incorrect ordering

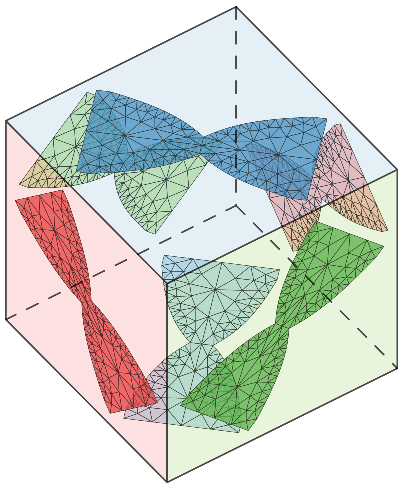

(b) Correct ordering

Figure 2: Plot of triangular mesh.

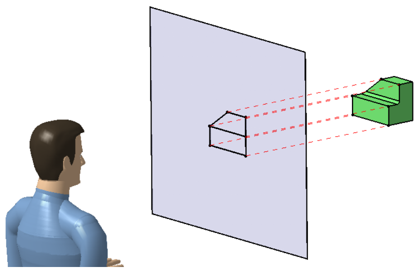


Figure 3: Orthogonal projection. Reprinted from https://intrinsys.com.

3 Criteria

3.1 General

- This project can be selected by unlimited number of students. However, no collaboration between students is expected.
- Project should be submitted including short documentation describing how the algorithm works.
- Like for regular projects, short presentation (couple of minutes) is expected.
- To be awarded with credits, it is enough to code a function dealing with

Tasks 1–3.

- To participate in the competition (and have a chance to get some awards), all technical criterias must be fulfilled. In that case, the computational time required to get the correct solution will be measured on reference PC (Win10 + up-to-date edition on MATLAB). Fist three student with the fastest codes will be awarded. List of awards is attached at the end of the document.
- It is possible to always withdraw from the competition and select of of regular projects. This decision should be discussed with lecturers and their approval is required.

3.2 Technical

- The main function must have following header:
  `[triangleOrder, triangelVisibility, shadowColor] = ...`
  `visibleTriangles(points, connectivityList, pointOfView, flag);`
- where:
  - `points`: Matrix with points coordinates, `double, [nPoints x 3]`.
  - `connectivityList`: Matrix with triangle vertexes referring to points matrix ,`double, [nTriangles x 3]`.
  - `pointOfView`: azimuth and elevation of camera position, `double, [2 x 1]`
  - flag distinguishing what should be the output:
    `flag = 'order'`, only order of triangles is required (default value)
    `flag = 'visibility'`, visibility of triangles is also required
    `flag = 'shadow'`, in case of bonus task.
  - `triangleOrder`: Vector of triangle orders containing numbers from 1 to `Ntriangles`, `double, [nTriangles x 1]`
  - `triangelVisibility`: Vector of visibility indicates containing values `1` for visible triangles, `0` for non-visible triangles and `2` for partly visible triangles, `double, [nTriangles x 1]`
  - `shadowColor`: intensity of shadow (values from scale 0 to 255), `double, [nTriangles x 1]`
- The function for computing cross-section area must have following header:
  `area = crossSecArea(points, connectivityList, pointOfView)`
- Testing meshes can be downloaded at course web page `http://cw.fel.cvut.cz/wiki/courses/a0b17mtb/projects/soutez`.
- Since the problem with ordering of triangles appears while exporting to ASCI file, an exporting function `mesh2txt` was prepared and minimal LaTeX script to plot exported mesh is available.

- It is possible to work with LaTeX part online at overleaf: https://www.overleaf.com/4846394315mswxbvtwmbmp. Please be aware this is a shared document for all participants, however you can clone it for your own purposes.
- The plot function should be implemented separately - will not be part of competition.
- The GUI is not required for this project.
- No toolboxes or external codes and libraries (dll, mex) are allowed.

4 List of Awards

Will be announced during the semester.

References

[1] Sagi Katz, Ayellet Tal, and Ronen Basri. "Direct Visibility of Point Sets". In: *ACM Trans. Graph.* 26.3 (July 2007).

[2] Wikipedia contributors. *Delaunay triangulation — Wikipedia, The Free Encyclopedia*. [Online; accessed 29-October-2018]. 2018. URL: https://en.wikipedia.org/wiki/Delaunay_triangulation.