

Parallel Computing for Computational Mechanics

Summer semester 2020

Homework 1

The subject of this homework is to read a mesh in the MIXD format and post-process it for visualization in Paraview. A skeleton code is provided in `hw1.tar.gz` (found on *RWTHmoodle*). The instructions on how to compile and execute the code can be found in the provided README file. Three meshes of different element size (coarse, fine, finest) are provided in the test folder. The instruction to parse the appropriate mesh input parameters (`settings.disk-coarse/fine/finest.in`) are also in the README file.

Information about the MIXD format:

- Files named `minf` store mesh information in ASCII.
- Files named `mxyz` store coordinates in binary (`sizeof(double)`).
- Files named `mien` store connectivity in binary (`sizeof(int)`).

The mesh can be read in C++ as follows:

```
#include <constant.h> // provided

main(int argc, char *argv)
{
    dummy = settings->getMxyzFile();
    file.open(dummy.c_str(), ios::in|ios::binary|ios::ate);

    readStream = new char [nsd*sizeof(double)];
    file.seekg (0, ios::beg);

    ...
    file.read(readStream, nsd*sizeof(double));
    swapBytes(readStream, nsd, sizeof(double));
    ...
    file.close();
}
```

Use the helper functions to put the mesh information in the right place in the code structure. The helper functions can be found in the header files (`*.h`). A description of the necessary functions is given hereafter:

- `getMinfFile()`: Return the path to the `minf` file.
- `getMxyzFile()`: Return the path to the `mxyz` file.

- `getMienFile()`: Return the path to the mien file.
- `node[i].setX(x = value)`: Set the x-coordinate value to “value” for node *i*.
- `node[i].setY(y = value)`: Set the y-coordinate value to “value” for node *i*.
- `elem[i].setConn(j, connValue)`: Set the connectivity for element *i*. Local node number *j* with global node index “connValue”.

After the mesh is read, the code will export it into a “*.pvtu” file. This file can be opened with Paraview.

Question:

1. In a typical implementation, each mesh edge is drawn multiple times. Can you think of an algorithm that would avoid that?

Turn in the program used to obtain the picture and a report (pdf file) containing:

- One picture per mesh with the edge visible
- The code used to obtain the picture
- The answer to the question

All this should be wrapped within a single tar file `hw1.tar.gz`, which can, e.g., be created by the command:

```
tar czvf hw1.tar.gz hw1/*
```

Please do not use any other name or format for the archive. Furthermore, the following guidelines apply:

- Solutions are accepted until April 17, 11:55 PM.
- change only the parts of the code labelled “// Add code here”.
- The solution to this project shall be done individually (not as a team) and handed in online by using the *RWTHmoodle* system.
- A file `hw1.tar.gz` containing the data can be found on the *RWTHmoodle* page.
- We encourage you to use the provided template to write your report (the report can be compiled by typing the command `make`), but this is not mandatory. You can create your own template or work with another program to type your report. The report must nevertheless include the required contents.

Contact:

Maximilian Schuster · schuster@cats.rwth-aachen.de

Violeta Karyofylli · karyofylli@cats.rwth-aachen.de