Prof. Dr. Christoph Pflaum
Benjamin Mann

Summer Term
2022

Simulation and Scientific Computing 2
Assignment 3

Interpolation between Unstructured Grids

Your task in this assignment is to write a program that carries out an optimized interpolation of scalar values from one unstructured grid to another. As explained in the lecture and the exercises, the naive way is rather time consuming, especially for larger grids, and the performance can be increased by introducing an additional structured grid. Furthermore, VTK legacy file reader and writer have to be implemented to read/write the grids.

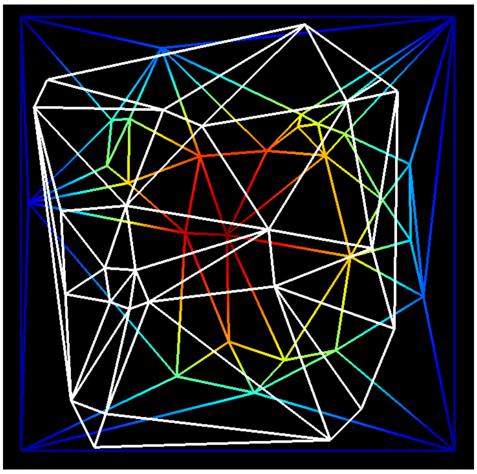

Figure 1: Exemplary visualization of the task. The source grid, `INGRID`, contains scalar values on each grid node (displayed as colored grid) which have to be interpolated to the destination grid, `OUTGRID`, (displayed white).

Tasks

1. You are given an `INGRID` and `OUTGRID` in the VTK format. You can visualize them using the open source visualization tool *Paraview*. In your program, read the data from the legacy VTK files[1] and store all necessary information in suitable data structures. As an example consider the following annotated VTK file:

```
# vtk DataFile Version 2.0     # file header
Sample triangulation           # this is only a comment
ASCII
DATASET UNSTRUCTURED_GRID

POINTS 4 float                 # 4 grid points
0.0 0.0 0.0                    # x, y and z coordinates of first grid point
1.0 0.0 0.0
1.0 1.0 0.0
0.0 1.0 0.0

CELLS 2 8                      # 2 cells (here triangles), need 8 information values
3 0 1 2                        # the 3 indices of corner nodes of first triangle
3 2 3 0

CELL_TYPES 2                   # type of the 2 cells
5                              # 5 denotes the vtk triangle type
5

POINT_DATA 4
SCALARS value float            # scalar field with identifier "value"
LOOKUP_TABLE default           # default color table
1                              # value of first grid point
2
3
4
...
```

2. Set up a structured grid with a uniform mesh size h. Choose the mesh size as $h = 1/\sqrt{N}$, where N is the number of grid points of `INGRID`. Also make sure that the structured grid fully covers the points of both grids, `INGRID` and `OUTGRID`, by determining the origin of the structured grid appropriately.

3. Store in each cell of this structured grid the indices of the triangles of `INGRID` that intersect with this cell. Since intersecting triangles with squares is a delicate task, you could also e.g. use the bounding box of the triangle instead which will introduce inaccuracies in this mapping.

4. Carry out the interpolation from the `INGRID` to the `OUTGRID` with the help of this structured grid. For the interpolation itself, use barycentric coordinates.

5. Write the `OUTGRID` together with the interpolated values as `OUTGRID_interpolated.vtk` to a VTK file in the legacy file format, as shown before.

[1] http://www.vtk.org/VTK/img/file-formats.pdf

6. Visualize both grids and their values in *Paraview* and check whether the interpolated values fit to the original values.

7. Different `INGRID`s and `OUTGRID`s are available on StudOn. First use the `const` version which features the constant value 1 and is thus well suited to test your implementation. Afterwards, use the others to check more complicated values.

8. Make sure you use double precision floating-point calculations.

9. Your program `intug` should be able to be executed with the following command:

   ```
   ./intug INGRID.vtk OUTGRID.vtk
   ```

   where `INGRID.vtk` is the grid containing scalar values on each grid node and `OUTGRID.vtk` is the grid to which you should interpolate these scalar values.

10. Please hand in your solution to this assignment until Thursday, **July 1, 2022** at 23:55! Make sure the following requirements are met:

    (a) The program should be compilable with a Makefile.

    (b) The program should compile without errors or warnings with the following g++ compiler flags:

    ```
    -O3 -Wall -Winline -Wshadow -std=c++17
    ```

    (c) The program should be callable as specified above and output the required values.

    (d) The solution should contain well commented source files, a fitting Makefile that satisfies all the conditions specified above, and instructions how to use your program (e.g. in form of a short README file).

    (e) Submit your solution on StudOn as a team submission and include all your team members. Make sure that all required files are included in your submission!

Credits

In this assignment the credits are awarded in the following way:

1. Up to five points are awarded if your program correctly performs the above tasks and fulfills all of the above requirements. Submissions with compile errors will lead to **zero** points!

2. Bonus task: One bonus point can be obtained for the following. Choosing the optimal mesh size h of the structured grid is hard to achieve and depends on the grids at hand. We want to benchmark the runtime for the finest `INGRID` and `OUTGRID` available and for different h. Use the following code fragment in your main function to measure the runtime of the interpolation:

   ```
   struct timeval t0, t;
   gettimeofday(&t0, NULL);

   # do the interpolation here

   gettimeofday(&t, NULL);
   std::cout << "Wall␣clock␣time␣of␣interpolation:␣" <<
   ((int64_t)(t.tv_sec - t0.tv_sec) * (int64_t)1000000 +
   (int64_t)t.tv_usec - (int64_t)t0.tv_usec) * 1e-3
   << "␣ms" << std::endl;
   ```

   Reading and writing the VTK files is excluded from the time measurements, but the construction of the structured grid has to be contained inside. Vary the mesh size $h = 1/2^i$, with $i \in \{0, 1, \ldots, 9\}$ and measure the runtime of your program. Plot the runtime (y-axis) over the mesh size (logarithmic x-axis) and explain the behavior.