Proseminar Physically-Based Simulation
Winter Semester 2017

Programming Assignment Sheet 3
Hand-out: November 30th, 2017
Hand-in: January 11th, 2018

2D Fluid Simulation

Outline

In this exercise, a rising smoke plume will be simulated on a 2D square grid. The implementation is inspired by the Stable Fluids approach of J. Stam (1999). Smoke is represented on the grid through scalar density values that are advected with the flow field. A source of smoke (i.e. density) is defined in the bottom region. Moreover, a user can add further smoke via mouse interaction (i.e. left mouse button). Simple buoyancy forces, proportional to density, are defined in Y-direction. Smoke dissipation, gravity forces, and viscosity-dependent diffusion are not included.

The main target of the assignment is to get acquainted with a simple fluid simulation, to implement the theoretical concepts taught in the lecture, and to test the performance of the numerical solution scheme.

The program provides a solution to the Euler Flow equations (i.e. reduced Navier-Stokes equations, omitting viscosity-dependent diffusion) for an incompressible, Newtonian fluid.

The target is to find a solution to the following 2nd order coupled partial differential equations

$$\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} = -\frac{1}{\rho}\nabla p + \mathbf{f}$$

$$\nabla \cdot \mathbf{v} = 0$$

with the problem domain given as the unit square

$$\Omega = [0,1] \times [0,1].$$

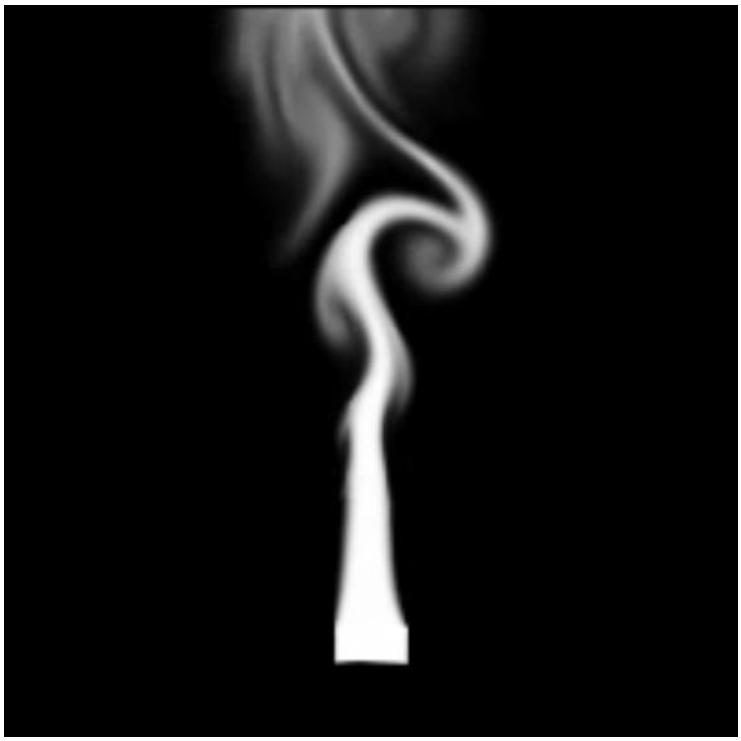
The domain is regularly subdivided into square cells. The step size is equal in X- and Y-direction. The resolution of the mesh (i.e. number of cells per each axis) can be specified as a command line parameter (default 200). Assume a constant density of the fluid of ρ=1.0 (i.e. close to air).

Boundary conditions are already implemented and can be changed by the user. Also, vorticity confinement is included and can be turned ON or OFF by the user. The differential equation problem is solved numerically via the splitting method. The key tasks are to implement two steps of this process – the Semi-Lagrangian advection and the pressure update.

Simulation Framework

A C++ project framework is provided that initializes simulation settings, executes the simulation loop, and uses Legacy OpenGL for visualization. Various options for user input exist (e.g. changing boundary conditions, enabling/disabling vorticity confinement, adding smoke density). Solving the assignment requires addition of code to the file `exercise.cpp`. It should not be required to change any other files; also no additional external libraries should be included.

After adding the required changes and afterwards running the code, the program should provide a smoke simulation similar to the following:



Tasks

The exercise consists of implementing parts of the numerical solution steps in a splitting approach for solving the Euler Flow equations. The code has to be added to the file `Exercise.cpp`.

1. Implement code for computing Semi-Lagrangian advection. Note that a regular grid is employed, i.e. it is not staggered. Both the density values as well as the fluid field itself (self-advection) have to be processed. External forces are updated before the advection step.

2. Implement a simple iterative solver for the Poisson equation, such as the Gauss-Seidel method. The maximum number of iterations is set to 1000, however, the iteration will be terminated earlier according to the following task.
   Compute the residual during the numerical iteration. Track convergence by examining the Euclidean norm of the residual vector. The error threshold for the iterative solver is given as 1e-5. Terminate the iteration when the residual is below this threshold.

3. As a final step, implement the velocity update to ensure a divergence-free velocity field. With this, the simple fluid solver should be functional (note that before no output will be visible).

Submission of your solution is due on January 11th, 2018 at 11:59pm. Please submit all source code (i.e. no executables) and report documents in a ZIP file via OLAT (submission access will be made available).

The solution should be carried out in teams of 2-3 students. Please get in touch if you are unable to find a team. Make sure to acknowledge all team members in the code and submission (please name the ZIP file with all surnames). Only one submission per team is required.

Finally, please respect the academic honor code. Each team should work on its own on the solution of the assignment. Any attempts at plagiarism or collusion will lead to 0 marks and further scrutiny. Please get in touch in case of questions or problems with regard to the assignment.

In total there are 20 marks achievable in this assignment.

Implementation Remarks

The solution should be developed using the provided framework. While you are free to develop in any environment, the final submission should compile and run in Linux. As reference, the machines in RR21 will be considered. Make sure that after unpacking your solution can be compiled directly in the top-level directory via `make`. Follow a proper coding style and provide appropriate comments in the code.