Vortexy fluid dynamics simulator

Roni Koitermaa *

February 18, 2020

Software documentation

*roninkoi@iki.fi

Contents

1 Introduction

Vortexy is a computational fluid dynamics (CFD) simulation package. It is written in C and uses the finite volume method with the SIMPLE algorithm to calculate flow of incompressible fluids, namely liquids.

The simulator is based on irregular tetrahedral meshes. These meshes can be computed from surfaces using the program Tetgen. The simulator takes a configuration file as input that contains paths to the simulation mesh and boundary conditions in addition to other settings. The state of the system is periodically written to an output file specified in the config. Included is also a renderer that uses OpenGL to visualize results.

2 Background

2.1 Navier-Stokes equations

The Navier-Stokes equations form the basis for all of fluid dynamics. The momentum equation is typically written as

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u} + g, \tag{1}$$

where $\mathbf{u} = (u, v, w)$ is velocity in [m/s], t time in [s], ρ density in [kg/m^3], p pressure in [Pa], $\nu = \frac{\mu}{\rho}$ kinematic viscosity in [m^2/s], g gravity in [m/s].

The continuity equation must be satisfied for incompressible fluids that have no sinks of sources

$$\nabla \cdot \mathbf{u} = 0, \tag{2}$$

where $\mathbf{u} = (u, v, w)$ is velocity in [m/s].

2.2 Turbulence

A simple way of predicting onset of turbulence is the Reynolds number:

$$\mathrm{Re} = \frac{\mu u L}{\rho} = \frac{uL}{\nu}$$

Turbulence models in simulations include RANS (Reynolds Averaged), LES (Large Eddy) and DNS (Direct).

2.3 Finite volume method

The *finite volume method* (FVM) is based on a simulation mesh with volume elements. This enables evaluation of partial differential equations (PDEs) prevalent in physics. The divergence theorem allows us to convert volume integrals to surface integrals

$$\int_V \nabla \cdot \mathbf{F} \, \mathrm{d}V = \oint_S \mathbf{F} \cdot \mathrm{d}\mathbf{S},$$

so volume terms can be computed from fluxes at element faces.

2.4 Discretization

Momentum equation:

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) = -\nabla p$$

Continuity equation:

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

A momentum matrix is constructed by decomposition:

$$M = A\mathbf{u} - H$$

Discretization of the momentum equation is achieved:

$$A\mathbf{u} - H = -\nabla p$$
$$\mathbf{u} = \frac{H}{A} - \frac{1}{A}\nabla p \tag{3}$$

Pressure equation:

$$\nabla \cdot \left[\left(\frac{1}{A} \right)_f \nabla p \right] = \nabla \cdot \left(\frac{H}{A} \right)_f \tag{4}$$

Continuity equation:

$$\sum_{f \in f_n} \dot{m}_f = 0$$

2.5 Gauss-Seidel method

2.6 SIMPLE algorithm

1. Set boundary conditions, set u and p
2. Compute gradients ∇u and ∇p
3. Under-relax momentum matrix M
4. Solve *momentum equation* for velocity guess u^*
5. Compute mass fluxes j_m, flow rate $\dot{m} = j_m \cdot A$
6. Solve *pressure equation*
7. Under-relax pressure
8. Correct mass fluxes j_m
9. Correct velocities u^{n+1}
10. (Update density ρ)
11. Goto 1 if not reached convergence

12. Increment time $t^{n+1} = t^n + \Delta t$

$$A\mathbf{u} = \begin{pmatrix} F & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ P \end{pmatrix} = \begin{pmatrix} f_b \\ 0 \end{pmatrix}$$

Equations written in form:

$$A\mathbf{u} + Bp = \mathbf{f}$$
$$C\mathbf{u} = 0, \quad \Big| \quad C = B^T$$

Gives Uzawa problem matrix (saddle point)

$$\begin{pmatrix} A & B \\ C & 0 \end{pmatrix}$$

2.7 Boundary conditions

Wall

Inlet

Outlet

3 Implementation

3.1 Simulation mesh

Triangular/tetrahedral, closed, connected, Delaunay

3.2 Solver

Gauss-Seidel

3.3 Rendering

OpenGL

4 Software

4.1 Compilation

Deps: glibc, (OpenGL, GLEW, GLFW)

```
cmake .
make
```

4.2 Configuration

Files: sim.cfg, data/fluid.cfg

4.3 Examples

4.4 Problems

Checkerboard problem

References

[1] https://quickersim.com/tutorial/tutorial-2-numerics-simple-scheme/

[2] https://www.openfoam.com/documentation/guides/latest/doc/guide-applications-solvers-simple.html

[3] https://www.cfd-online.com/Wiki/SIMPLE_algorithm