Contents

1. Simulations (10 pgs)

This part of thesis serves as a widen documentation for the `PyVort` codebase, a new platform to simulate quantum vortex rings. The code is written in well commented Python 3, arranged in a modular structure. The primary aim of this documentation is to highlight which module(s) are involved and how they work. In appendix, one can find a table of the parameters (user's options) which can be set in the `config.py` file.

To run the code, there has to be installed only Python 3 (with various libraries) on any OS. All the Python code can be found in the directory `src/`. The entire project is open-source and can be found as a public GitHub repository. Pull requests of any further development would be definitely appreciated.

At present the code can be run only using infinite boundary conditions. Therefore, only closed-loop vortices can be realized using simulation. However, the codebase flexibly supports the implementation of on-closed loops.

1.1 Vortex filament model

The `PyVort` code is based on vortex filament (VF) model, a technique pioneerd by Schwarz in the early 1980s. Superfluid vortex filament is represented by a series of mesh points distributed along the centerline of the VF. The motion of the whole VF is summed up by the motion of each mesh point.

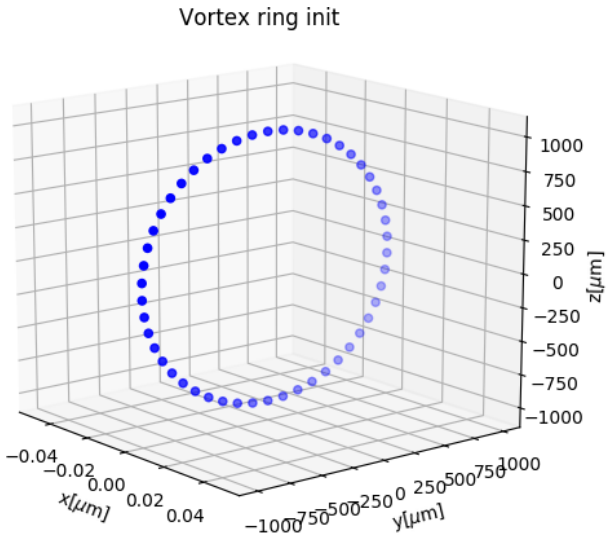

Figure 1.1: Example of vortex ring segments

More precisely we define the VF as a three dimensional curve $\mathbf{s}(\xi, t)$. Here ξ represents arc-lengths and t is time. Each mesh point is given by its coordinates $\mathbf{s}_i$, direct neighbour indices (previous $(i-1)$ and next mesh point $(i+1)$). This resolves in a directed digraph, which is a good starting point for the initial data structure.

We can construct the tangent vector $\mathbf{s}'$, then normal vector $\mathbf{s}''$, and the binormal vector $\mathbf{s}' \times \mathbf{s}''$ by taking numerical derivatives. Note $\mathbf{s}' = \mathrm{d}\mathbf{s}/\mathrm{d}\xi$, and so on.

Finite differences

Numerical derivatives $\mathbf{s}'$ and $\mathbf{s}''$ need to be properly calculated. At a particular segment with position $\mathbf{s}_i$, we define the distance to the particle in-front $\mathbf{s}_{i+1}$ as $l_i = |\mathbf{s}_i - \mathbf{s}_{i+1}|$ and the distance to the particle behind $\mathbf{s}_{i-1}$ as $l_{i-1} = |\mathbf{s}_{i-1} - \mathbf{s}_i|$. By in-front/behind we refer to the particles next/previous along the filament. Similarly, we can define the l_{i+1} and l_{i-2} for the farther particles.

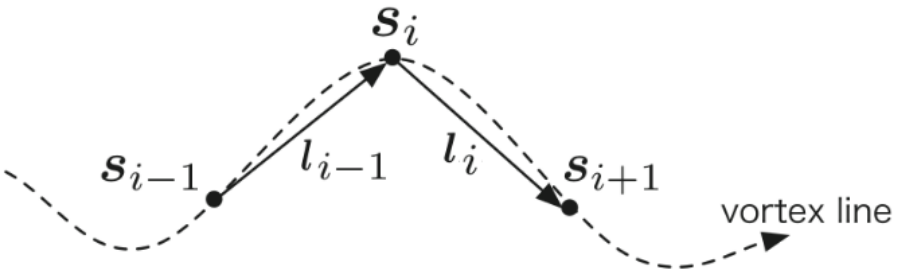

Figure 1.2: Depiction of a few segments and corresponding lengths

For accuracy, we approximate all spatial derivatives $\mathbf{s}'_i$, $\mathbf{s}''_i$ by a fourth-order finite difference method (FD), which can also account the varying distances along the vortex filament Using FD theorem, we can construct the approximations by taking the Taylor's series expansions. We can then write:

$$\frac{\mathrm{d}^n \mathbf{s}_i}{\mathrm{d}\xi^n} \approx A_i \mathbf{s}_{i-2} + B_i \mathbf{s}_{i-1} + C_i \mathbf{s}_i + D_i \mathbf{s}_{i+1} + E_i \mathbf{s}_{i+2} \qquad \text{for } n \in \{1, 2\} \tag{1.1}$$

Calculation of coefficients A, B, C, D, E can be done in many ways. In code, we use both the analytical solution (closed form) and the solution by inverting the Vandermonde matrix. The first one is obviously faster, but the second one is more reliable.

Biot-Savart discretisation

We denote the external sources of velocity fields as $\mathbf{v}_{n,ext}$ and $\mathbf{v}_{s,ext}$. The equation of motion is then given directly by Schwarz's equation(**??**):

$$\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}t} = \mathbf{v}_{s,ext} + \mathbf{v}_i + \mathbf{v}_{\mathrm{mutual}} \tag{1.2}$$

The first difficulty in the VF model come from the calculation of term $\mathbf{v}_i$. As we shown in (**??**), this advection term can be split into the LIA part and Biot-Savart integral:

$$\mathbf{v}_{\mathrm{ind}}^{(i)} = \frac{\varkappa}{4\pi}(\mathbf{s}' \times \mathbf{s}'') \ln\left(\frac{2\sqrt{l_{i-1}l_i}}{a}\right) + \frac{\varkappa}{4\pi}\int_{\mathcal{L}'} \frac{(\mathbf{r}' - \mathbf{s}) \times \mathrm{d}\mathbf{r}'}{|\mathbf{r}' - \mathbf{s}|^3}\,, \tag{1.3}$$

where l_{i-1} and l_i are the arc lengths of the curve between points $\mathbf{s}_{i-1}$ and $\mathbf{s}_i$ and between $\mathbf{s}_i$ and $\mathbf{s}_{i+1}$, and $\mathcal{L}'$ is the original vortex line without the section between $\mathbf{s}_{i-1}$ and $\mathbf{s}_{i+1}$.

Due to our discretisation, the Biot-Savart integral can be rewritten as the sum of line contributions between each j and $j+1$ segments (except for the ones attached to the i-th point):

$$\mathbf{v}_{\mathrm{BIOT}}^{(i)} \approx \sum_{j \notin \{i-1,i\}} \frac{\varkappa}{4\pi} \frac{(R_j + R_{j+1})(\mathbf{R}_j \times \mathbf{R}_{j+1})}{R_j R_{j+1}(R_j R_{j+1} + \mathbf{R}_j \cdot \mathbf{R}_{j+1})}\,, \tag{1.4}$$

where $\mathbf{R}_j = \mathbf{s}_j - \mathbf{s}_i$ and $\mathbf{R}_{j+1} = \mathbf{s}_{j+1} - \mathbf{s}_i$.

State definition

In Python 3, the single vortex ring object is represented using `class` structure. This structure is partially updated after each step, so the *state* of the vortex ring is defined with following properties:

- shape - a dictionary of three parameters: ring center coordinates $[x_c, y_c, z_c]$, radius R and the direction of desired motion $\{x, y, z\}$ (three possible axis)
- velocity - the actual velocity of vortex ring center $\mathbf{v}_c$
- number of segments - number of segments N the vortex ring is composed of
- segments - a whole array of particular segments with following attributes:

- coordinates - an array of segment coordinates $[x_i, y_i, z_i]$
- previous/next neighbour - array indices of the *previous* $(i-1)$ and the *next* $(i+1)$ segment from the geometrical point of view
- tangent/curvature - a tangential and normal vectors $\mathbf{s}_i'$ and $\mathbf{s}_i''$
- LIA velocity - a self-induced velocity $\mathbf{v}_{\mathrm{LIA}}(\mathbf{s}_i)$ driven by the near surrounding.
- BIOT velocity - a self-induced velocity driven by the farther parts of the vortex ring $\mathbf{v}_{\mathrm{BIOT}}(\{\mathbf{s}_j\})$
- Drive velocity - a velocity given by the mutual friction $\mathbf{v}_{\mathrm{drive}}(\mathbf{s}_i)$
- Full velocity - the sum of external sources $\mathbf{v}_{s,ext}$, LIA velocity, BIOT velocity and the (quantum) drive velocity

1.2 Integration

- Euler vs. RK4 step
- time stepping
- stability

1.3 Resegmentation

- adding and removing segments
- local spline

Points along the line are added (or removed) if the vortex is stretched (or compressed).

1.4 Vortex ring

- initialisation
- movement, decreasing radius
- comparison with theory
- Kelvin waves (?)

1.5 Future implementations

If any two lines become very close (a distance less than the separation along the line) then the filaments reconnect, changing the topology of the system.