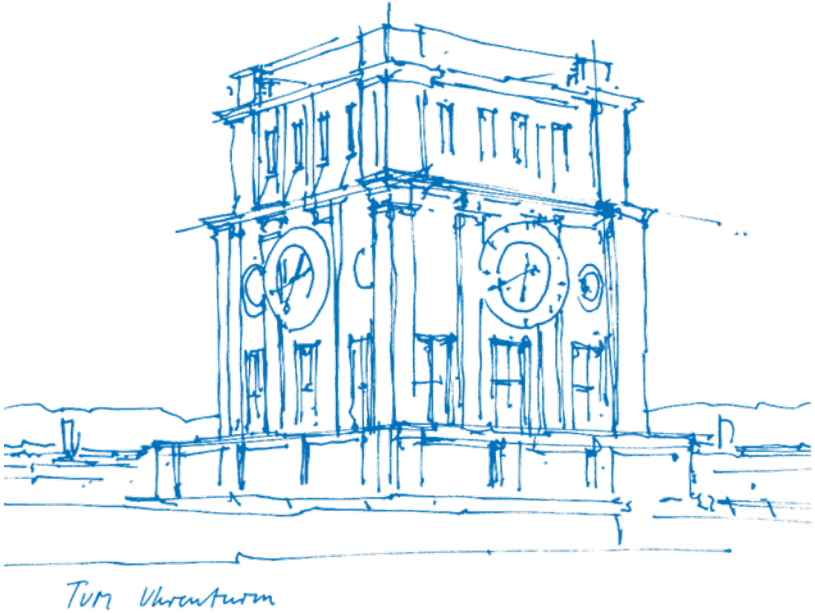
Molecular Dynamics - Assignment 3

Alex Hocks   Jan Hampe   Johannes Riemenschneider
Technische Universität München
TUM CIT
Lehrstuhl für wissenschaftliches Rechnen

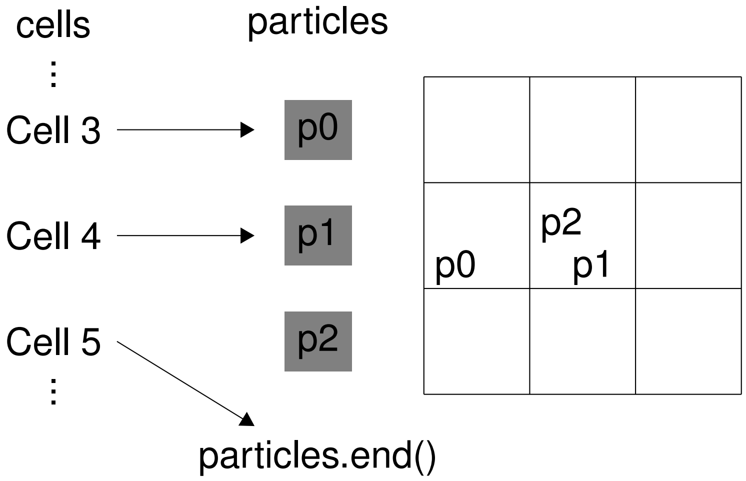
7. Dezember 2022



The Cell Data-Structure - Approach 1

Idea:

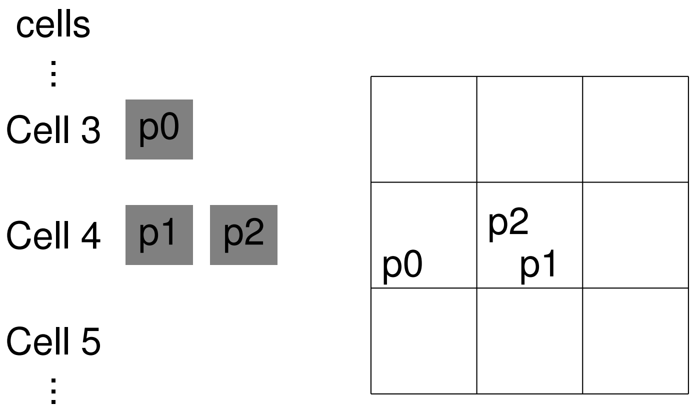
- Sort Particles in accordance to their Cell Position
- save which part of the particles-Vector corresponds to which cell



The Cell Data Structure - Approach 2

Idea:

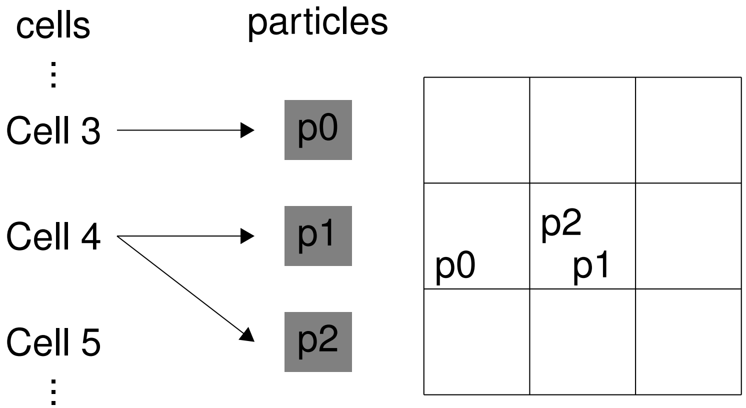
Approach 1.1 stored multiple virtual vectors in one vector → let's actually store the particles in vectors corresponding to their cell



The Cell Data Structure - Approach 3

Idea:

- Each Cell only keeps references to their members
- No sorting or copying of entire particles required



Approach Comparison

Approach 1	Approach 2	Approach 3
+ Easy to implement	+ Easy to implement	+ Easy to implement
+ Interface for old Assignments remains unchanged	+ New Implementation of some methods needed	− Interface for old Assignments remains unchanged
− Expensive struct swaps during sorting	− Expensive struct copies with potential reallocs needed	+ References are cheap
+ Direct access to particles for calculations	+ Direct access to particles for calculations	+ Dereferencing needed

Approach Comparison

Approach 1	Approach 2	Approach 3
+ Easy to implement	+ Easy to implement	+ Easy to implement
+ Interface for old Assignments remains unchanged	+ New Implementation of some methods needed	− Interface for old Assignments remains unchanged
− Expensive struct swaps during sorting	− Expensive struct copies with potential reallocs needed	+ References are cheap
+ Direct access to particles for calculations	+ Direct access to particles for calculations	+ Dereferencing needed

In the end we decided to implement approach 3. ¸

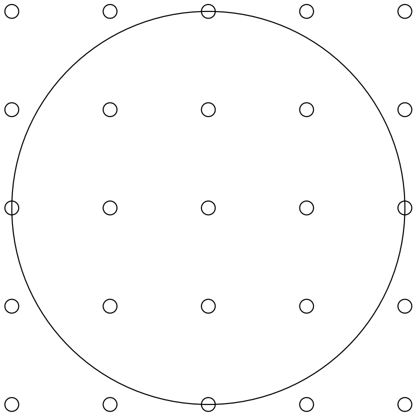
Spheres

Expansion of the Body-struct utilized in Assignment 2

$$\sqrt{x^2+y^2+z^2} <= r$$

$$\iff x^2+y^2+z^z <= r^2$$

```
enum Shape {cuboid, sphere};
struct Body {
        Shape shape;
        Eigen::Vector3d fixpoint;
        Eigen::Vector3d dimensions;
        double distance;
        double mass;
        Eigen::Vector3d start_velocity;
} ;
```

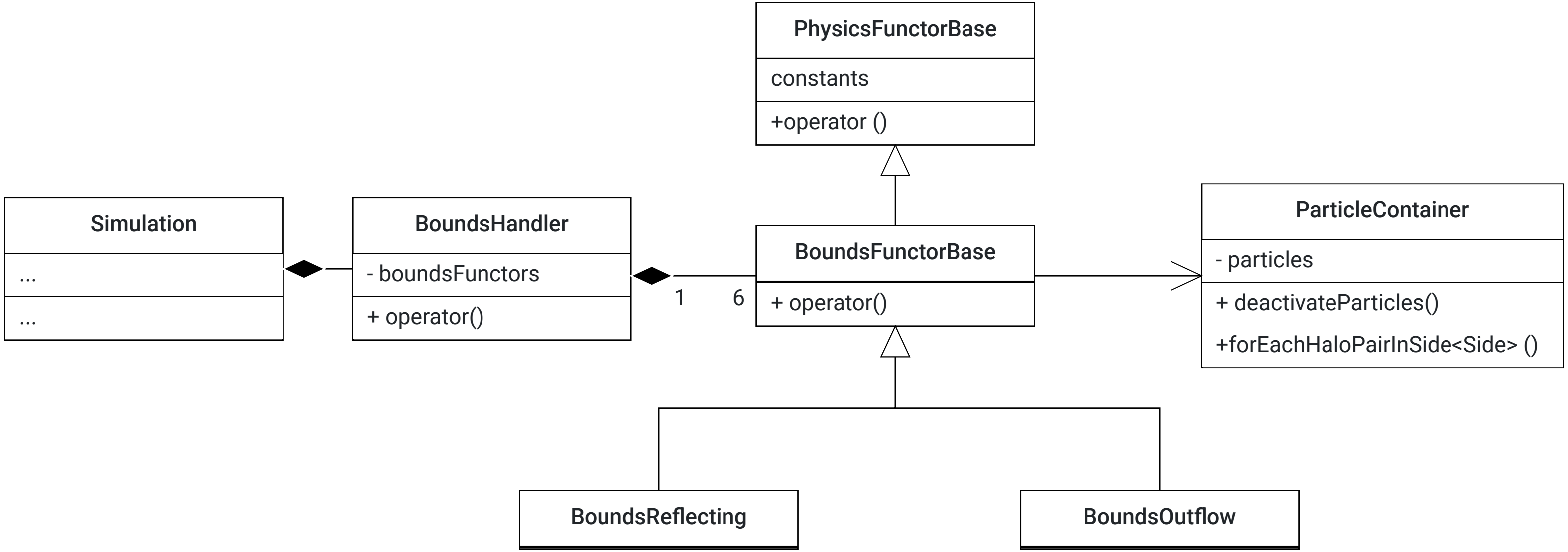


ParticleContainer’s new methods

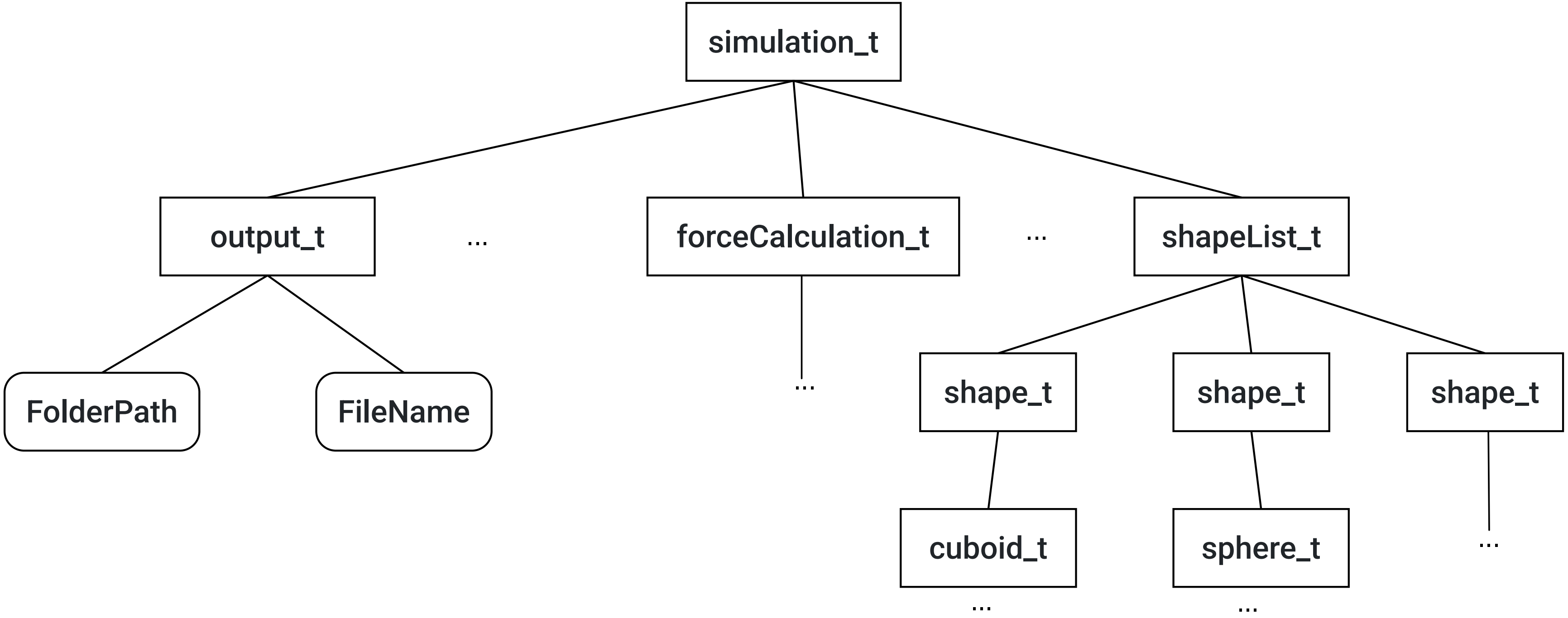
```
class ParticleContainer{
        ...

        void forAllPairsInSameCell(void (*fun) (Particle& p1, Particle& p1));

        void forAllPairsInNeighbouringCell(void (*fun) (Particle& p1, Particle& p1));

        void forAllCells(void (*fun)(...));

        void forAllDistinctNeighbouringCells(void (*fun) (...));

        ...
}
```

- Functionality of first two methods is sufficient but hard to optimize
- Functionality of last two methods results in higher cohesion, but potential for runtime improvement

Bounds Handling



XSD- tree-like definition of Datastructures



XSD- Code Snippet

- simulation_t stores all the necessary parameters
- utilizes defined other tree-like Datastructures
- “simulation_t is root“

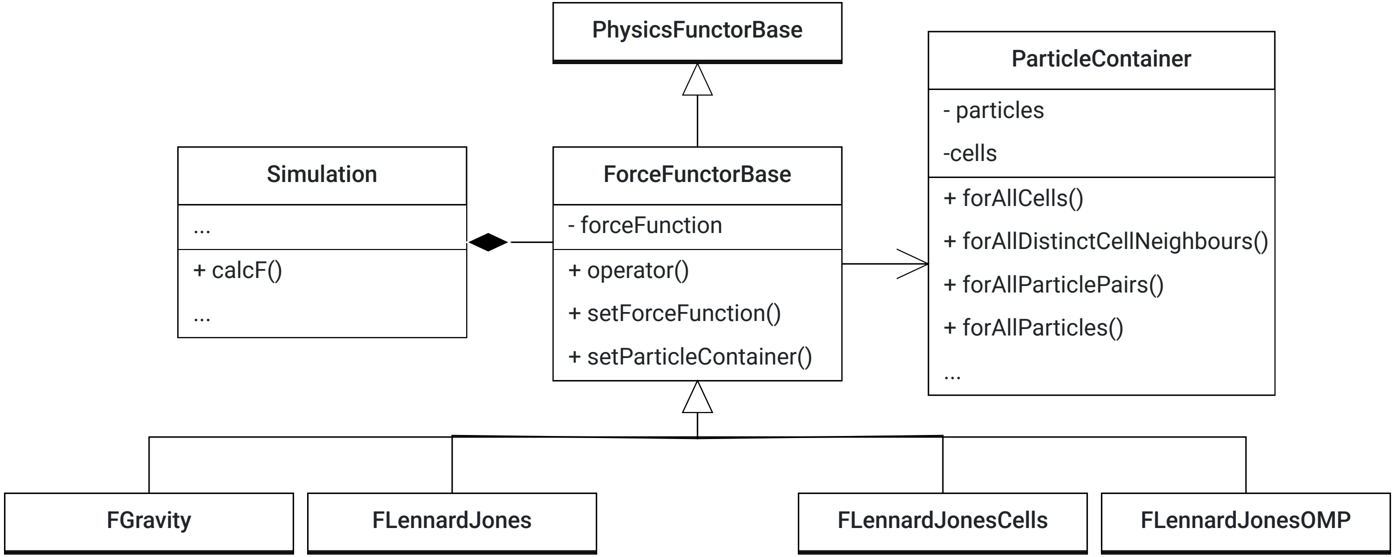
```
<xsd:complexType name="simulation_t">
        <xsd:sequence>
                <xsd:element name="OutputFile" type="output_t" minOccurs="0"/>
                ...
                <xsd:element name="ShapeList" type="shapeList_t"/>
        </xsd:sequence>
</xsd:complexType>
```

XSD- Code Snippet

```
<xsd:complexType name="shape_t">
        <xsd:choice>
                <xsd:element name="Particle" type="particle_t"/>
                <xsd:element name="Cuboid" type="cuboid_t"/>
                <xsd:element name="Sphere" type="sphere_t"/>
        </xsd:choice>
</xsd:complexType>
```

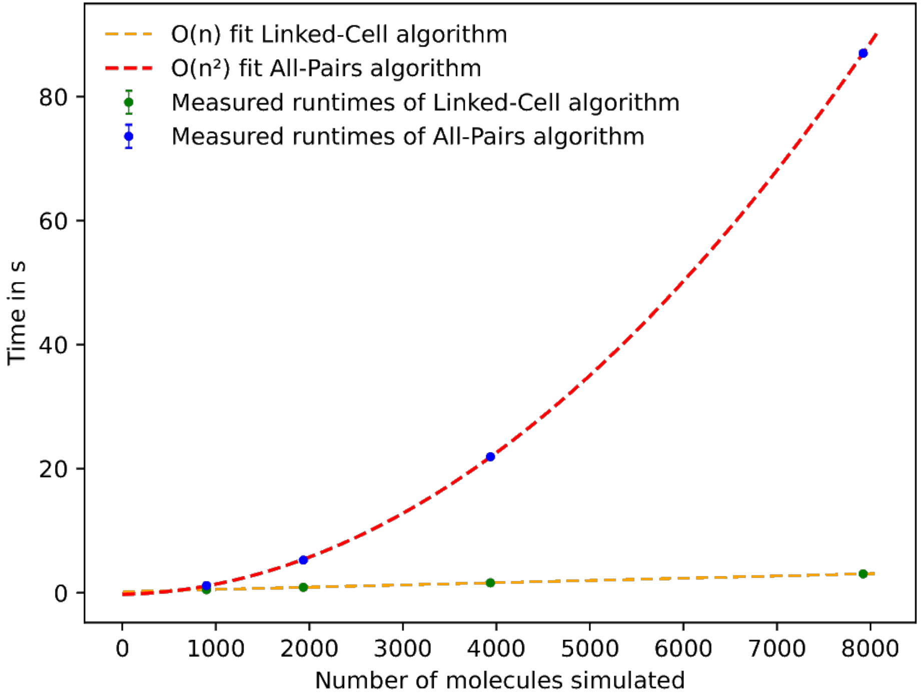
ForceFunctors

- Force function used gets determined on runtime via input parameters
- Force functor defines the algorithm (Linked-Cell algorithm/ All-Pairs algorithm) used



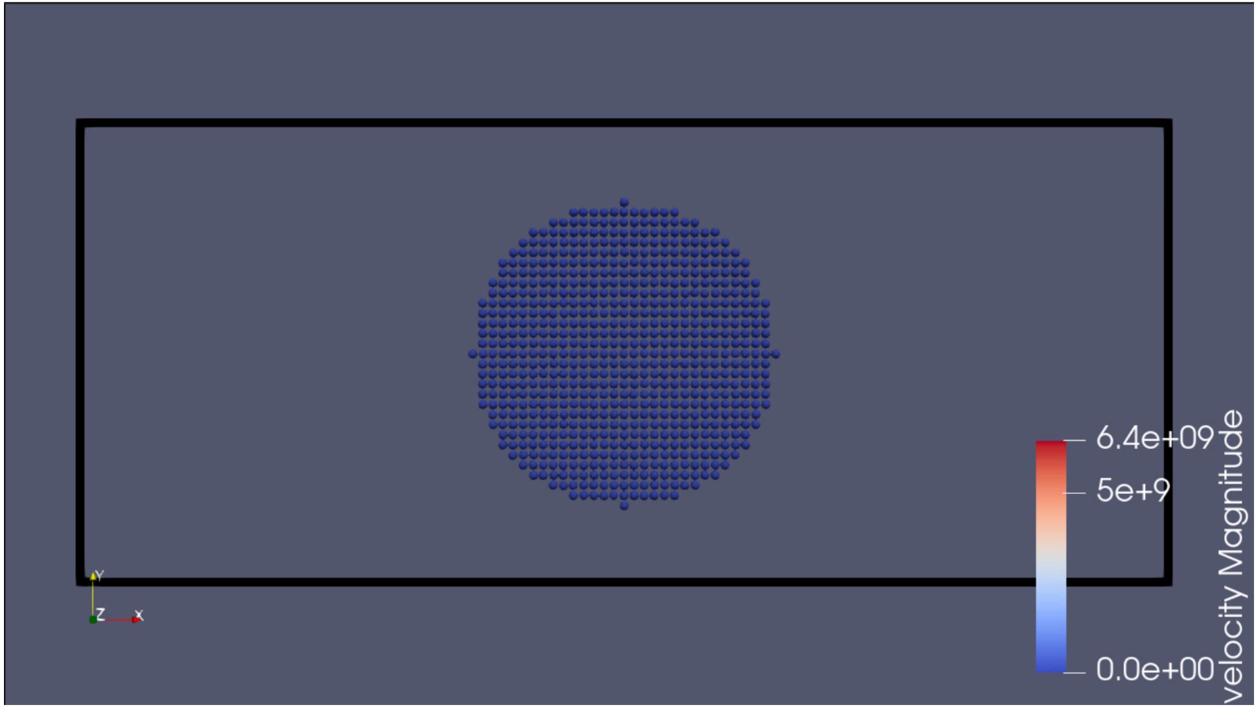
Benchmark setup

Runtime Comparison of different algorithms

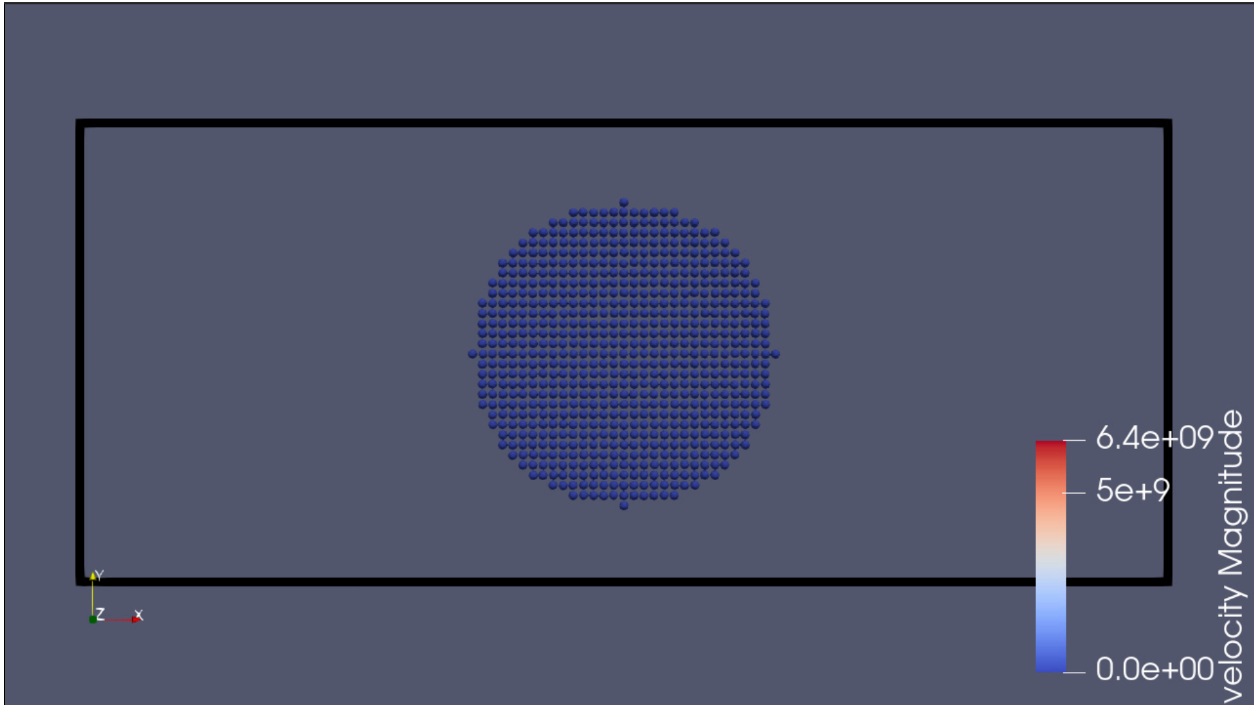


Ubunutu 20.04 LTS
i7 –12700KF @5, 0GHz
64GB RAM @ 3200MHz

Our Simulation



Doubling Δt


Y
Z
X
6.4e+09
5e+9
0.0e+00
velocity Magnitude