Explosions

Computational Physics Presentation

Ville, Aapo, Hannu

April 28, 2015

Introduction

- A nice group project? $\rightarrow$ Do something completely different than during the course
- Idea: simulate explosions - a non-trivial and fairly interesting subject
- The actual implementation studies effectively an explosion of gas particles in a vacuum
- Unrealistic? No - very similar to the intuitive conception of explosions
- In an explosion the density of the gas that explodes is much higher than that of the surroundings
- Only when the explosive gas has expanded enough, the surrounding density becomes important - and this regime is not of interest here

Physics of an explosion

- High density, pressure or temperature with respect to surroundings $\rightarrow$ explosion
- Start with a high-density gas cloud $\rightarrow$ the physics of the system will take care of the rest
- Required: fluid dynamics for compressible systems
- A shock wave should be produced to the progressing explosion front
- Handling of dynamic length scales: begin with a compressed system and end with an expanded system
- Only the dynamical properties of the system are studied, so temperature does not require explicit handling

Fluid simulation

- Simulating the behaviour of a fluid is highly non-trivial
- Handling a quickly expanding fluid is a non-trivial case within the group of fluid simulations
- A solution from astrophysics: smoothed particle hydrodynamics
- Astrophysics is familiar with changing length scales and vacuum boundary conditions - this fits well a simulation of explosions
- The basic idea is to model the fluid with a group of discrete particles
- Fluid physics are obtained by applying a smoothing kernel function to the particles

SPH basics

- Dirac delta function $\delta(\mathbf{x})$ can be approximated using a kernel function $W(\mathbf{x}, h)$ (for instance Gaussian)
- A function can be represented using delta functions $\rightarrow$

$$f(\mathbf{x}) = \int_V f(\mathbf{x}')\delta(\mathbf{x}-\mathbf{x}')d\mathbf{x}' \approx \int_V \frac{f(\mathbf{x}')}{\rho(\mathbf{x}')} W(\mathbf{x}-\mathbf{x}', h)\rho(\mathbf{x}')d\mathbf{x}' \tag{1}$$

- Approximate integral over the density with a sum over point masses:

$$f(\mathbf{x}) \approx \sum_i m_i \frac{f(\mathbf{x}_i)}{\rho_i} W(\mathbf{x} - \mathbf{x_i}, h) \tag{2}$$

- Now for instance the density can be calculated:

$$\rho(\mathbf{x}) \approx \sum_i m_i W(\mathbf{x} - \mathbf{x_i}, h) \tag{3}$$

Smoothing kernel functions

- Normalized and delta-like, $\int W(\mathbf{r}, h) d\mathbf{r} = 1$
- Gaussian form would be nice, but it goes to zero only at infinite distance
- We use a commonly utilized cubic spline form that goes to zero at the distance of $2h$
- Generally, W uses $x = |\mathbf{x} - \mathbf{x}'|/h$ as a parameter and is proportional to $1/h^d$
- In an explosion the system expands much and it is essential to have adaptive h values
- At particle locations $\rho h^d =$ const. is desirable

SPH fluid equations

- Taking the time derivate of the discrete density turns out to give a discretized version of the continuity equation
- Using a hydrodynamical Lagrangian, an equation of motion can be constructed that functions effectively as the Euler equation
- Similarly the internal energies could be updated if the temperatures were to be studied
- An artificial vicous term can be introduced, for instance $\Pi \propto v^2/\rho$ - special form for HE simulations
- This upgrades the Euler equation to handle better for instance shock fronts (broaden the shock across several h)
- In a sophisticated implementation also the growth of entropy would be taken into account

Recap. of necessary equations

- The equation of motion is essential:

$$\mathbf{a}_i = -\sum_{j\neq i} m_j \left(\frac{P_j}{\rho_j^2} + \frac{P_i}{\rho_i^2} + \Pi_{ij} \right) \nabla_i W_{ij} \tag{4}$$

- The densities at particle locations are obtained as before:

$$\rho_i = \sum_j m_j W_{ij} \tag{5}$$

- The pressures are obtained from the ideal gas eos.:

$$P = k\rho^\gamma \tag{6}$$

- Updating h follows from $\rho h^d = \text{const.}$:

$$\frac{dh_i}{dt} = -\frac{h_i}{\rho d} \sum_{j\neq i} m_j (\mathbf{v}_i - \mathbf{v}_j) \cdot \nabla_i W_{ij} \tag{7}$$

Basic implementation

- Begin with a 2D implementation, easy to move to 3D
- Loop over timesteps and keep updating particle locations, velocities and h values
- To obtain the changes for $\mathbf{x}$, $\mathbf{v}$ and h densities and pressures need to be calculated
- An important sub-task is determining the significant neighbors of a particle - these are at most on the distance $2h$
- A naive implementation goes through all pairs and has $O(N^2)$ complexity, which can be improved
- EOM. can be handled with the leap-frog algorithm (or e.g. RK)
- Updates of h can be stabilized with relaxation or with hard reset of the constant value

Leap-Frog algorithm

- The idea is to stabilize the updating schemes of velocities and locations by updating velocities and locations that are a half timestep apart
- This can be expressed so that both the velocities and densities are stored on the same timestep, but the algorithm still visits half-timestep values:

$$\mathbf{v}_{i+1/2} = \mathbf{v}_i + \frac{\Delta t}{2}\mathbf{a}_i \tag{8}$$

$$\mathbf{r}_{i+1/2} = \mathbf{r}_i + \frac{\Delta t}{2}\mathbf{v}_i \tag{9}$$

- Full steps:

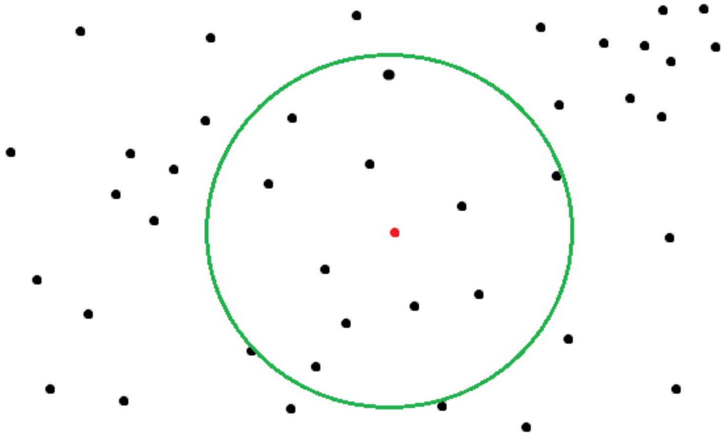
$$\mathbf{v}_{i+1} = \mathbf{v}_i + \Delta t\mathbf{a}_{i+1/2} \tag{10}$$

$$\mathbf{r}_{i+1} = \mathbf{r}_i + \frac{\Delta t}{2}(\mathbf{v}_i + \mathbf{v}_{i+1}) \tag{11}$$

Neighbor list handling

- Simple case $O(N^2)$: for each particle loop through particles that are after it in a list - each pair handled only once
- Not all particles are used for calculating system properties (e.g. densities) at a certain location
- When iterating though particles, distances between the current “center” and other particles are calculated and too distant particles do not contribute

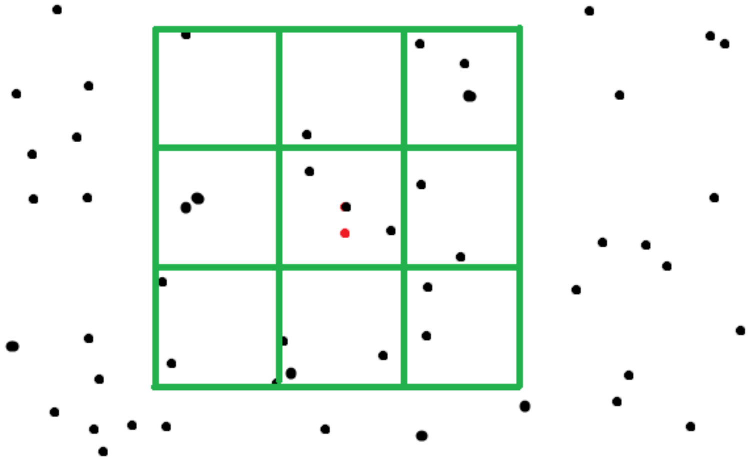
Neighbor list handling



Neighbor list handling

- The distance cutoff for the simple case leads to an intuitive alternative
- Compartmentalizing $O(N)$: Form a grid, and create and maintain a list of particles inside each grid box. Then for each particle, find the particle lists of the neighboring grid boxes.
- Use these lists to solve relevant system properties.

Neighbor list handling



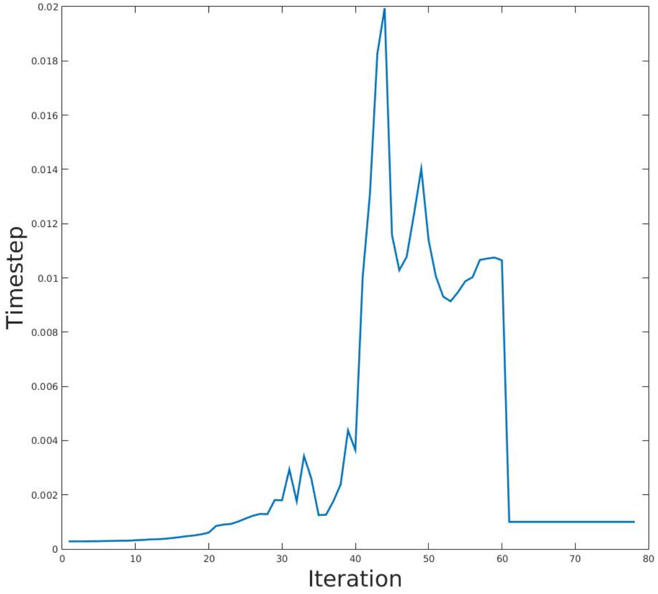
Timestep optimization

- Relatively small timesteps are informative, but if the steps are very small the simulation times become lengthy
- The density and velocities of the particles change during the simulation $\rightarrow$ the optimal timestep?
- Numerical stability and accuracy vs information lag between the particles $\rightarrow$ global timestep and lockstep evolution (here only CFL and force condition)

$$\delta t_i = \min(\delta t_{\mathrm{CFL,i}}(h), \delta t_{\mathrm{F,i}}(h), \delta t_{\mathrm{RKF,i}}, \delta t_{\mathrm{z,i}}) \tag{12}$$

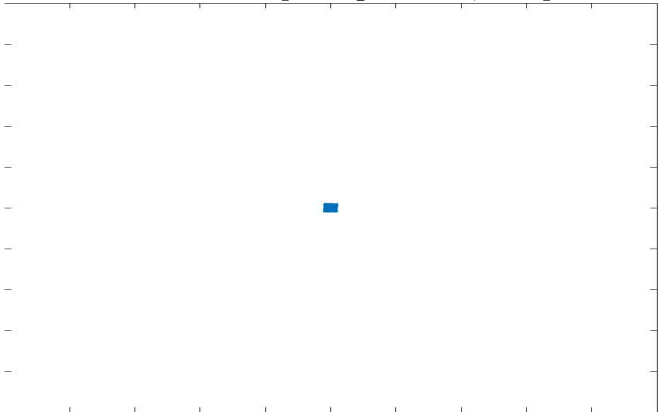
$$\delta t_{\mathrm{glob}} = \min_i \delta t_i \tag{13}$$

- Physical meaning: eg. CFL criterion, spatial information transfer speed $<$ local sound speed


Timestep
Iteration
0.02
0.018
0.016
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
10
20
30
40
50
60
70
80

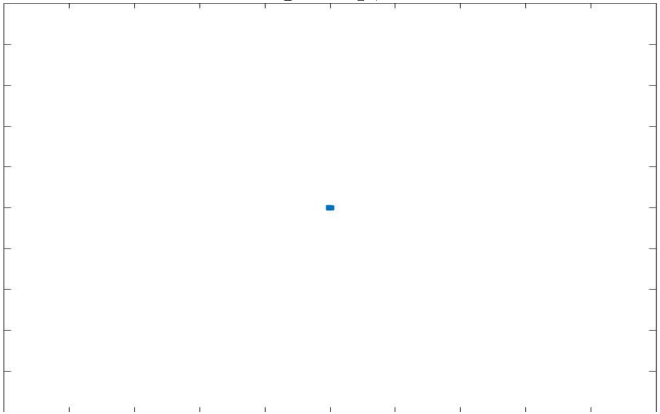
First 2D results

- Basic case: random square gas cloud, 300 particles



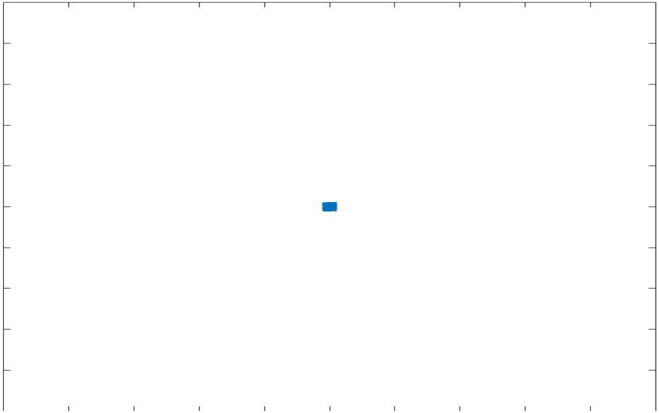
Dense 2D

- 4 times denser starting setup, 10 times slower time



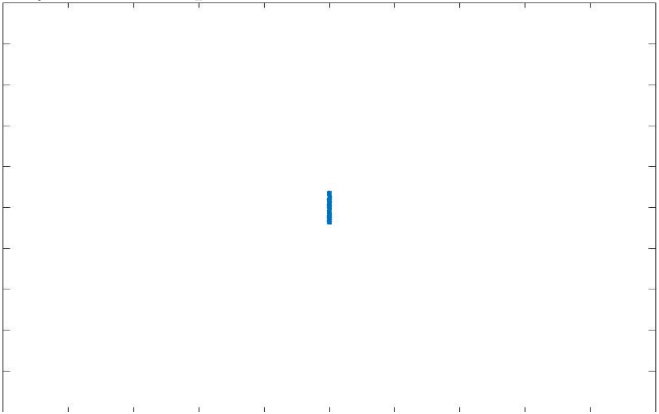
Dense 2D

- 4 times more massive starting setup, 10 times slower time



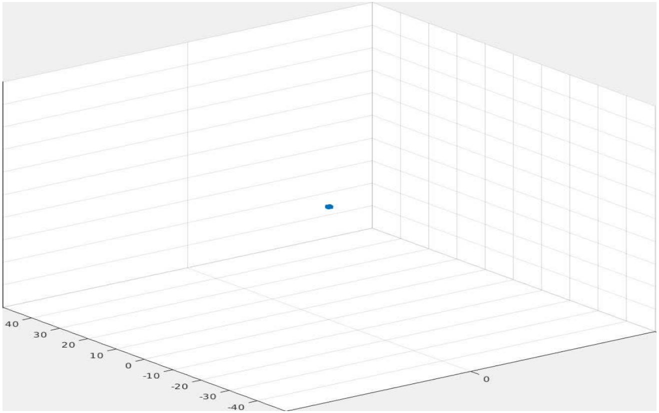
Asymmetric 2D

- Asymmetric explosion



3D

- Explosion in 3D implemented with very little changes



Conclusions

- Huehuehue

References

- P. J. Cossins, Smoothed Particle Hydrodynamics, arXiv:1007.1245v2 [astro-ph.IM], 2010
- G. R. Liu and M. B. Liu, Smoothed particle hydrodynamics - a meshfree particle method, World Scientific Publishing, Singapore, 2003