Games and Simulation

2020-2021

Fernando Birra
Rui Nóbrega

Simulation

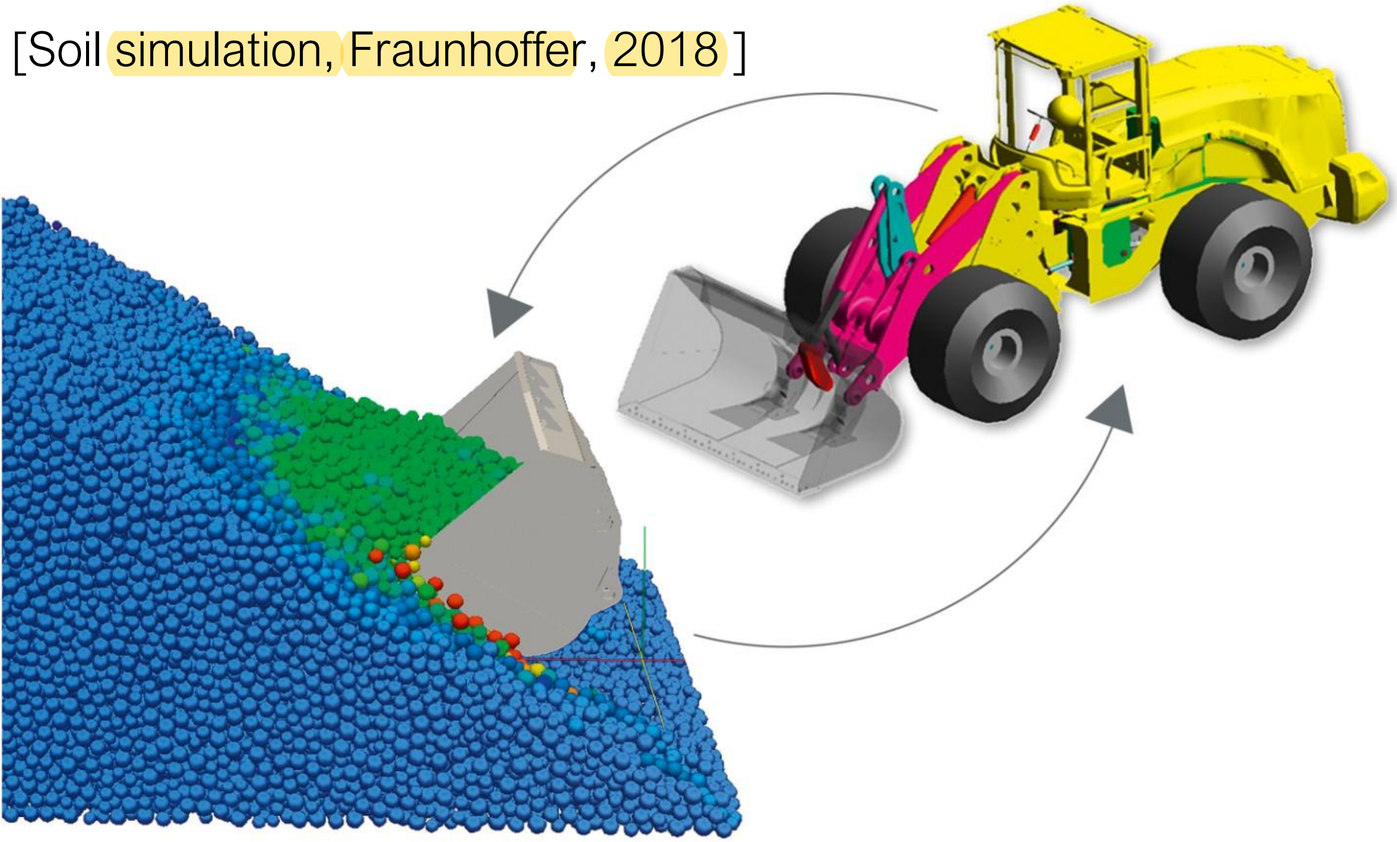
Fernando Birra
Rui Nóbrega

Particle Simulation

- Important tool for Scientific Simulation in Chemistry, Physics, Engineering etc.
- Introduced in the 50s by Alder and Wainwright with the initial *hard spheres model* and liquid simulation.
- In 1964 Rahman simulated liquid argon.
- Simulations: Water (1974), proteins (1977).
- Can be used for DNA, molecules, lipid systems, spatial trajectories, asteroid simulation etc.

Particle Simulation

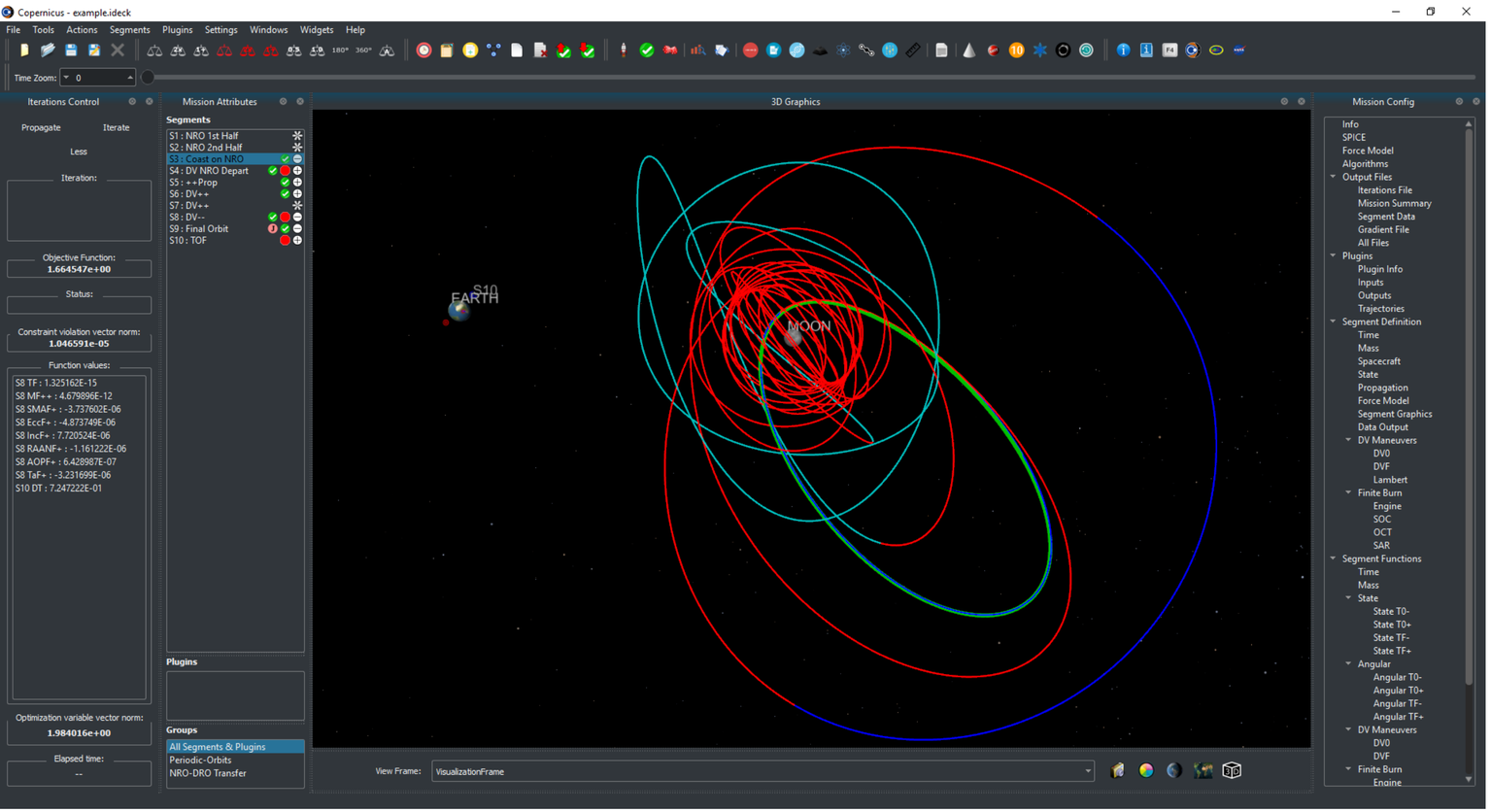
[Soil simulation, Fraunhoffer, 2018]

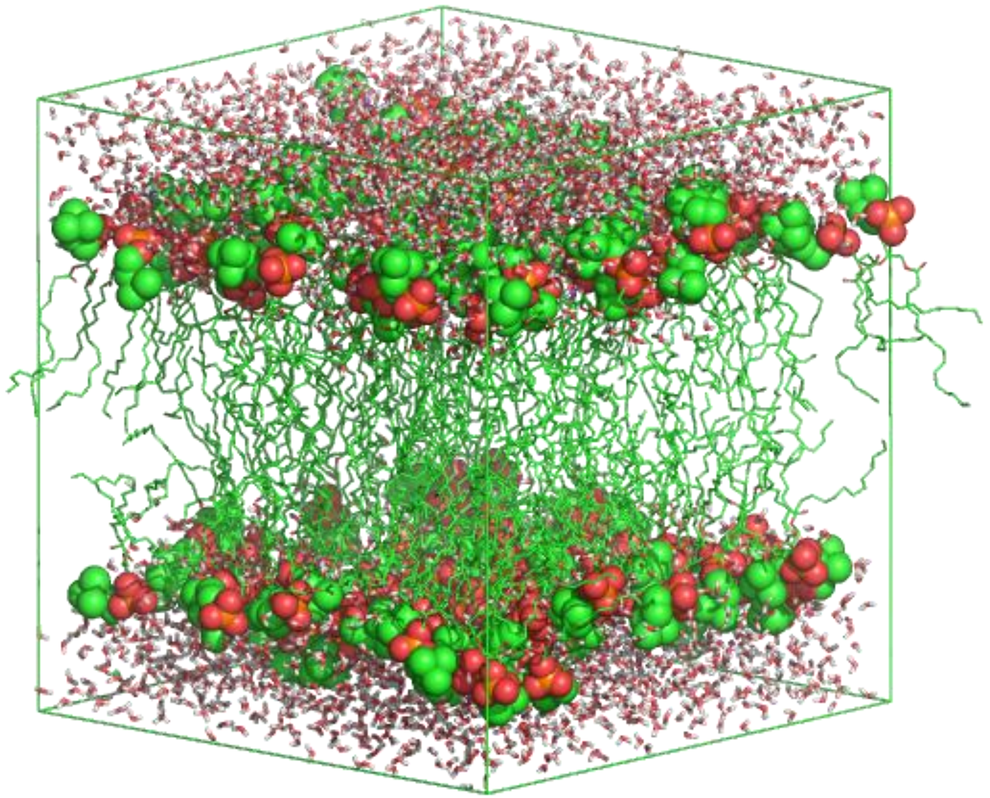


[Particle Simulation, Laura Remler]

Particle Simulation

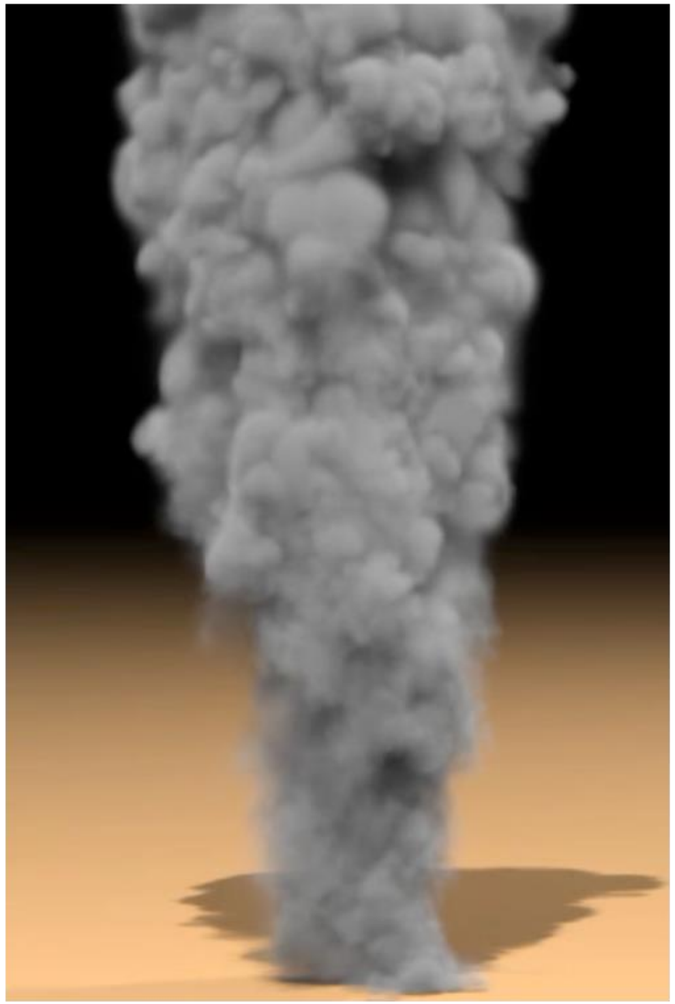
[Copernicus Trajectory Design, NASA]

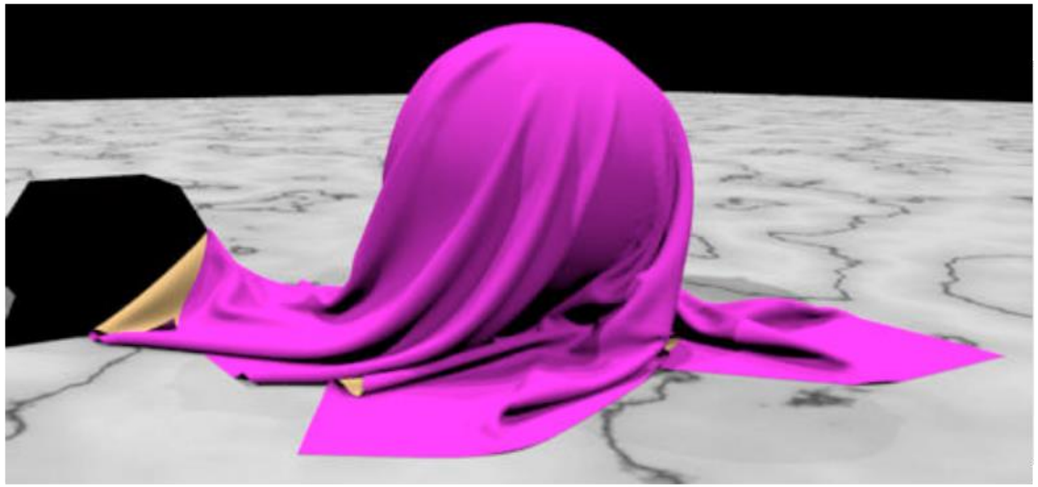


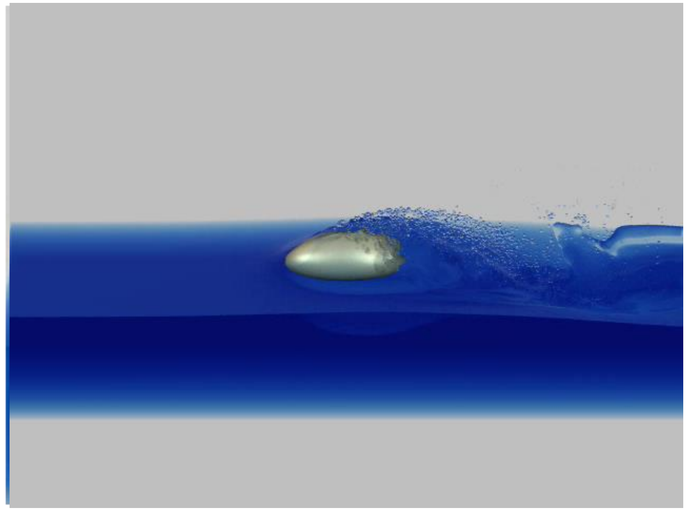


[Lipid bilayer simulation, Aponte-Santamaría, 2012]

Particle Simulation







[Smoke, Cloth, Hair, Water simulations]

Particle Systems Simulation

$$X_i = \begin{bmatrix} x \\ x' \\ \dots \\ x^N \\ f \\ m \end{bmatrix}$$

- Every particle interacts with every other particle
- The interaction is modulated by **forces**: gravity, springs …
- The **forces** depend on the *value x* and *1st*, *2nd*, 3rd .. derivatives over *Time*

- In a Newtonian Particle the **Force** depends on:
  position (value) , *velocity* (1st derivative), *acceleration* (2nd derivative) over *Time*

Newtonian Particle P_i

- $F = ma$ where $F(p, v, t)$ depends on the position, velocity and time.
- $x' = v$
- $v' = a = f/m$
- For each particle P_i we will keep:
  - $\overrightarrow{p_i}$ Position and $\overrightarrow{v_i}$ Velocity vectors – 2D, 3D or *n*D vector
  - $\overrightarrow{f_i}$ Force Accumulator vector – sum of all the forces from other particles that affect this particle
  - m mass

$$X_i = \begin{bmatrix} x \\ x' \\ \dots \\ x^N \\ f \\ m \end{bmatrix} \Rightarrow \quad P_i = \begin{bmatrix} p \\ v \\ f \\ m \end{bmatrix}$$

Particle System S

- A particle system is characterized by having n particles and a simulations time t.
- The time between each simulation step is Δt.
- $S = [particles \quad n \quad t]$

$$\begin{bmatrix} p_1 & p_2 & \cdots & p_n \\ v_1 & v_2 & \cdots & v_n \\ f_1 & f_2 & \cdots & f_n \\ m_1 & m_2 & \cdots & m_n \end{bmatrix}$$

Particle System S

- $S = [particles \quad n \quad t]$

$$\begin{bmatrix} p_1 & p_2 & \cdots & p_n \\ v_1 & v_2 & \cdots & v_n \\ f_1 & f_2 & \cdots & f_n \\ m_1 & m_2 & \cdots & m_n \end{bmatrix}$$

- For each particle:
  - Calculate all forces based on 2nd, 3rd … derivatives (example in gravity model $a = G$)
  - Update velocities
  - Update positions.

Particle System Simulation

```
simulation( System S, Δt, steps)
{
    foreach(step in steps){
        clearAllForces( S );


    }
}
```

$$\begin{bmatrix} p_1 & p_2 & \cdots & p_n \\ v_1 & v_2 & \cdots & v_n \\ 0 & 0 & \cdots & 0 \\ m_1 & m_2 & \cdots & m_n \end{bmatrix}$$

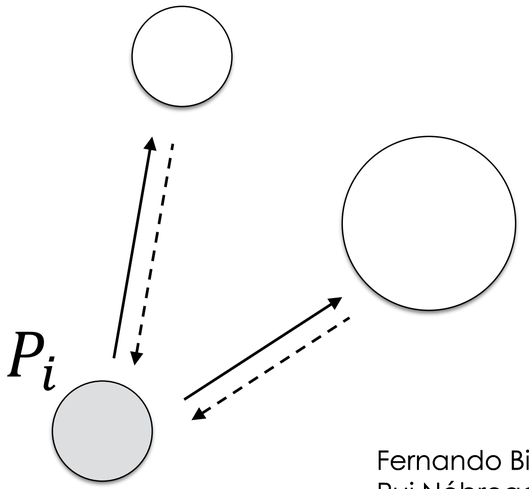
Particle System Simulation

```
simulation( System S, Δt, steps)
{
    foreach(step in steps){
        clearAllForces( S );
        calculateAllForces( S );

    }
}
```

$$\begin{bmatrix} p_1 & p_2 & \cdots & p_n \\ v_1 & v_2 & \cdots & v_n \\ f_1 & f_2 & \cdots & f_n \\ m_1 & m_2 & \cdots & m_n \end{bmatrix}$$

- For every particle calculate the force create by other particles.
- Sum all the forces that affect each particle.



Particle System Simulation

```
simulation( System S, Δt, steps)
{
    foreach(step in steps){
        clearAllForces( S );
        calculateAllForces( S );
        calculateAllVel&Pos( S, Δt );
    }
}
```

$$\begin{bmatrix} p_1 & p_2 & \cdots & p_n \\ v_1 & v_2 & \cdots & v_n \\ f_1 & f_2 & \cdots & f_n \\ m_1 & m_2 & \cdots & m_n \end{bmatrix}$$

- For every particle calculate new velocity and new position, based on the acceleration extracted from the accumulated force $a = f/m$ and the time Δt.

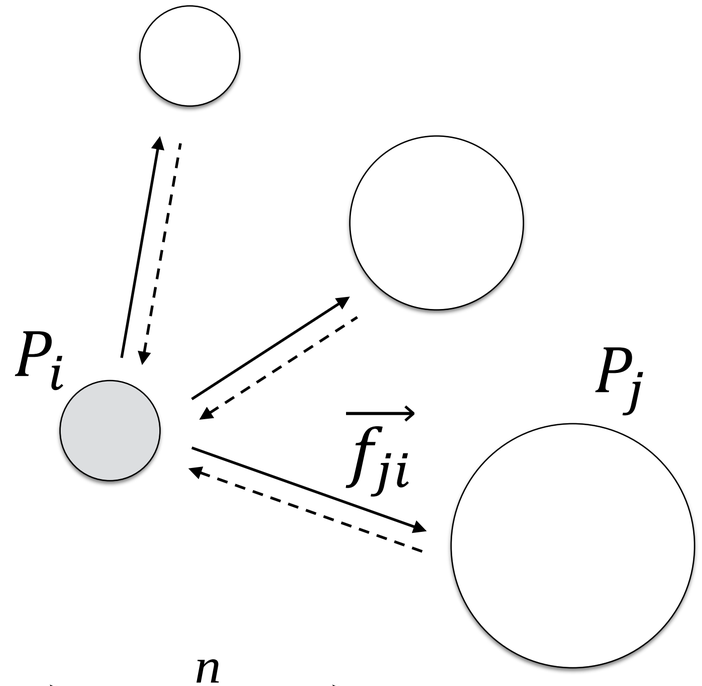
Calculate all Forces

```
calculateAllForces( S );
```

For each particle *i* we calculate the accumulated force f_i exerted by all other particles.

Note that according to Newton's 3rd Law forces between two objects exist in equal magnitude and opposite direction.

This reduces the amount of forces to be calculated in half.

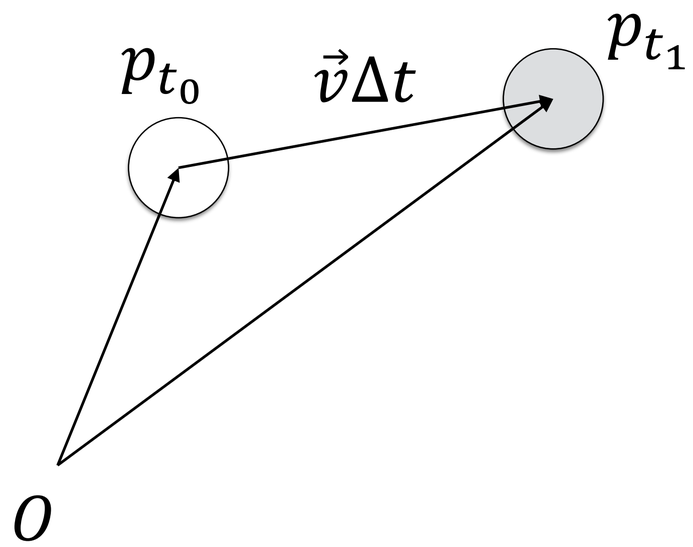


$$\overrightarrow{f_i} = \sum_{\substack{i=1 \\ j \neq i}}^{n} \overrightarrow{f_{ij}}$$

$$\overrightarrow{f_{ji}} = -\overrightarrow{f_{ij}}$$

Calculate Velocities and Positions

`calculateAllVel&Pos( S, Δt );`



For each particle *i* we use the previously calculated force to calculate the variation in velocity and position and update the v and p vectors.

Example using an Euler Interpolator:

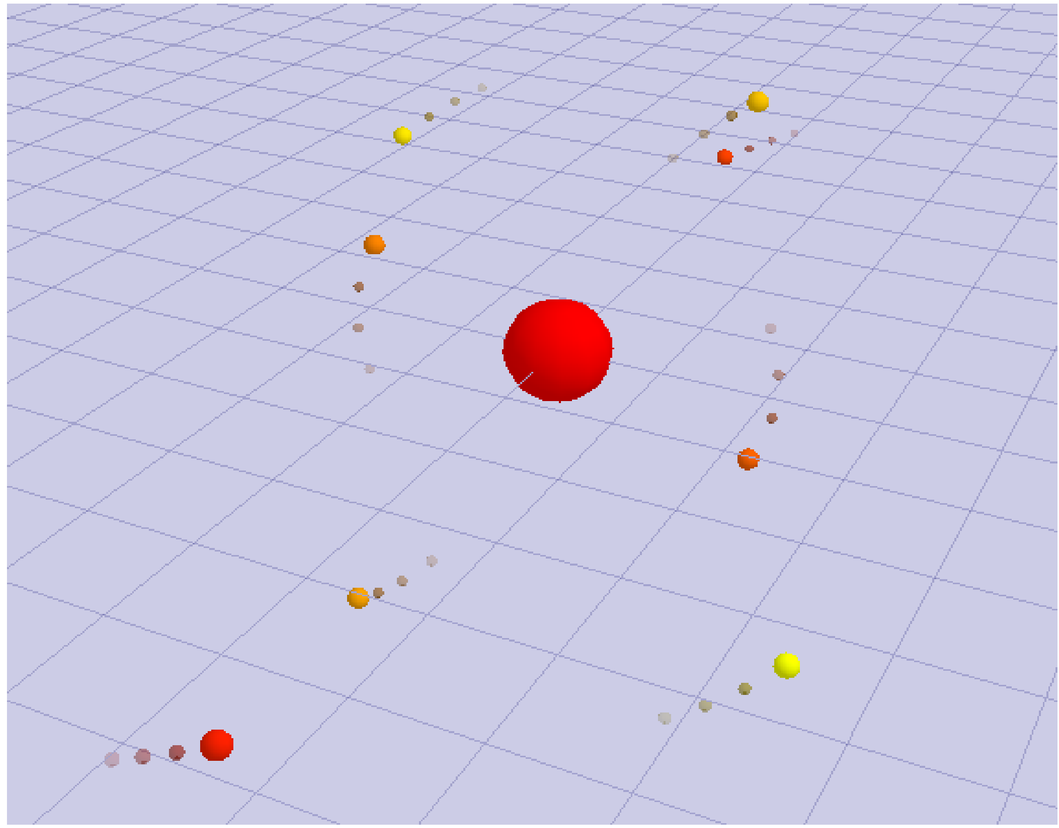
$$\Delta\vec{v}_i = \frac{\Delta t}{m_i}\vec{f}_i$$

[Euler Interpolator]

$$\Delta\vec{p}_i = \vec{v}_i\Delta t$$

Particle Systems Parameters

- In this particle systems simulation, there are several possible parameters:
  - **Force Model**: how should we calculate $\overrightarrow{f_{ij}}$?
  - **Interpolation Model**: how should we calculate $\Delta\overrightarrow{v_i}$ and $\Delta\overrightarrow{p_i}$?
  - **Collision Model**: how should we calculate collisions?

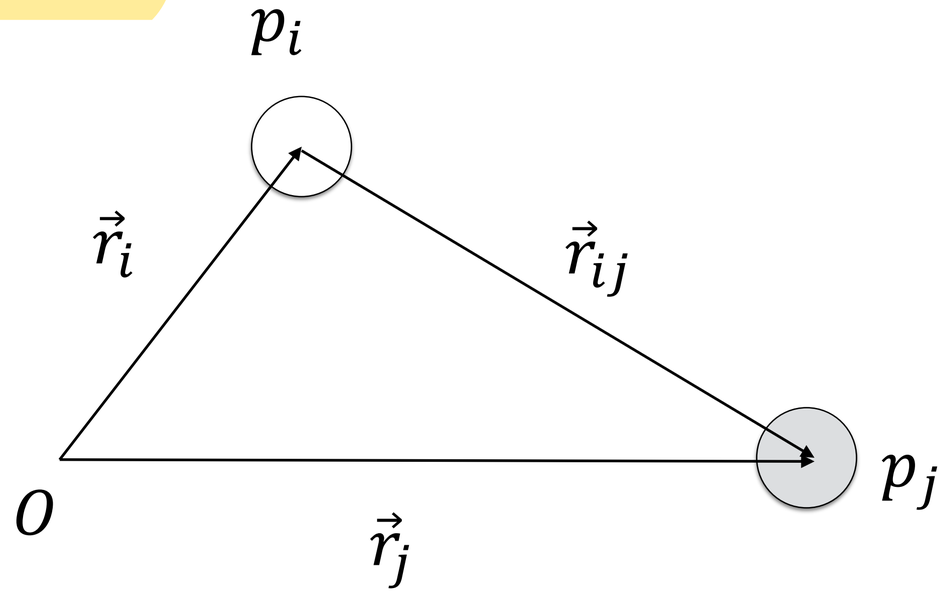


Particle Systems Parameters

- In this particle systems simulation, there are several possible parameters:
  - **Force Model**: how should we calculate $\overrightarrow{f_{ij}}$?
  - **Interpolation Model**: how should we calculate $\Delta\overrightarrow{v_i}$ and $\Delta\overrightarrow{p_i}$?
  - **Collision Model**: how should we calculate collisions?

- Gravity model,
  Hooke's model,
  Lennard-Jones's model,
  Hard Spheres Model
- Euler,
  Verlet
- Hard Spheres

Force Models



- How to calculate $\vec{f}_{ij}$?
  - (eq1) $\vec{d} = \vec{r}_{ij} = \vec{r}_j - \vec{r}_i$
  - (eq2) $\vec{f}_{ij} = \|\vec{f}_{ij}\|\overrightarrow{u_{ij}}, where\ u\ is\ a\ unit\ vector$
  - (eq3) $\vec{f}_{ij} = \|\vec{f}_{ij}\| \frac{\vec{d}}{\|\vec{d}\|}, force\ in\ a\ certain\ direction$
  - (eq4) $\vec{f}_{ij} = -\vec{f}_{j\mathrm{i}}$

Force Models: Gravity Model

- The Gravity Model uses the G the gravitic constant ($6.6720 \times 10^{-23} Nm^2 Kg^{-2}$) to create a force of attraction between particles.

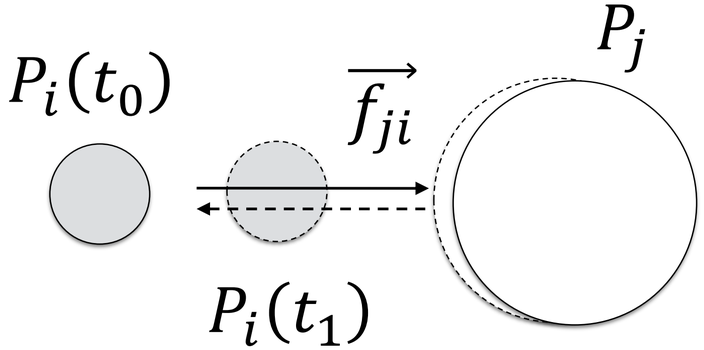
- Larger when the objects are close or the masses are larger.

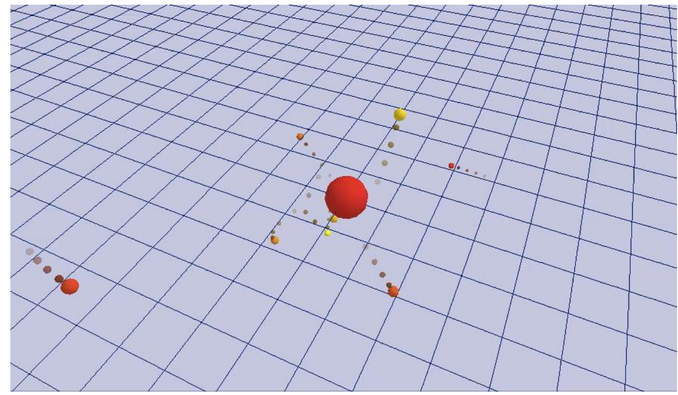
- (eq5) $\|\vec{f}_{ij}\| = \frac{G m_i m_j}{\|\vec{d}\|^2}$

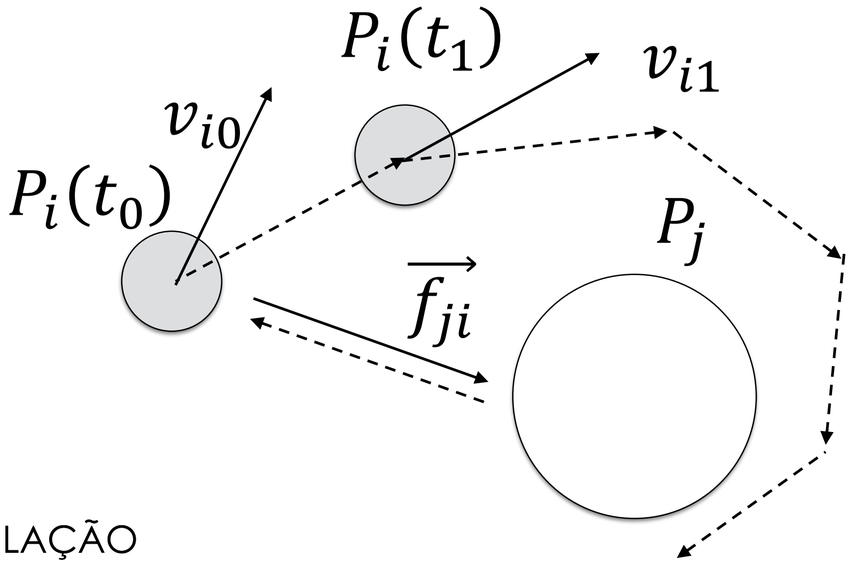
- Using (eq3) we get the vectorial form (eq6) $\vec{f}_{ij} = \frac{G m_i m_j}{\|\vec{d}\|^3} \vec{d}$

Force Models: Gravity Model

- (eq6) $\vec{f}_{ij} = \frac{G m_i m_j}{\|\vec{d}\|^3} \vec{d}$
- In a system without initial velocities the particles collapse
- When the velocity is perpendicular to the force, an orbit emerges





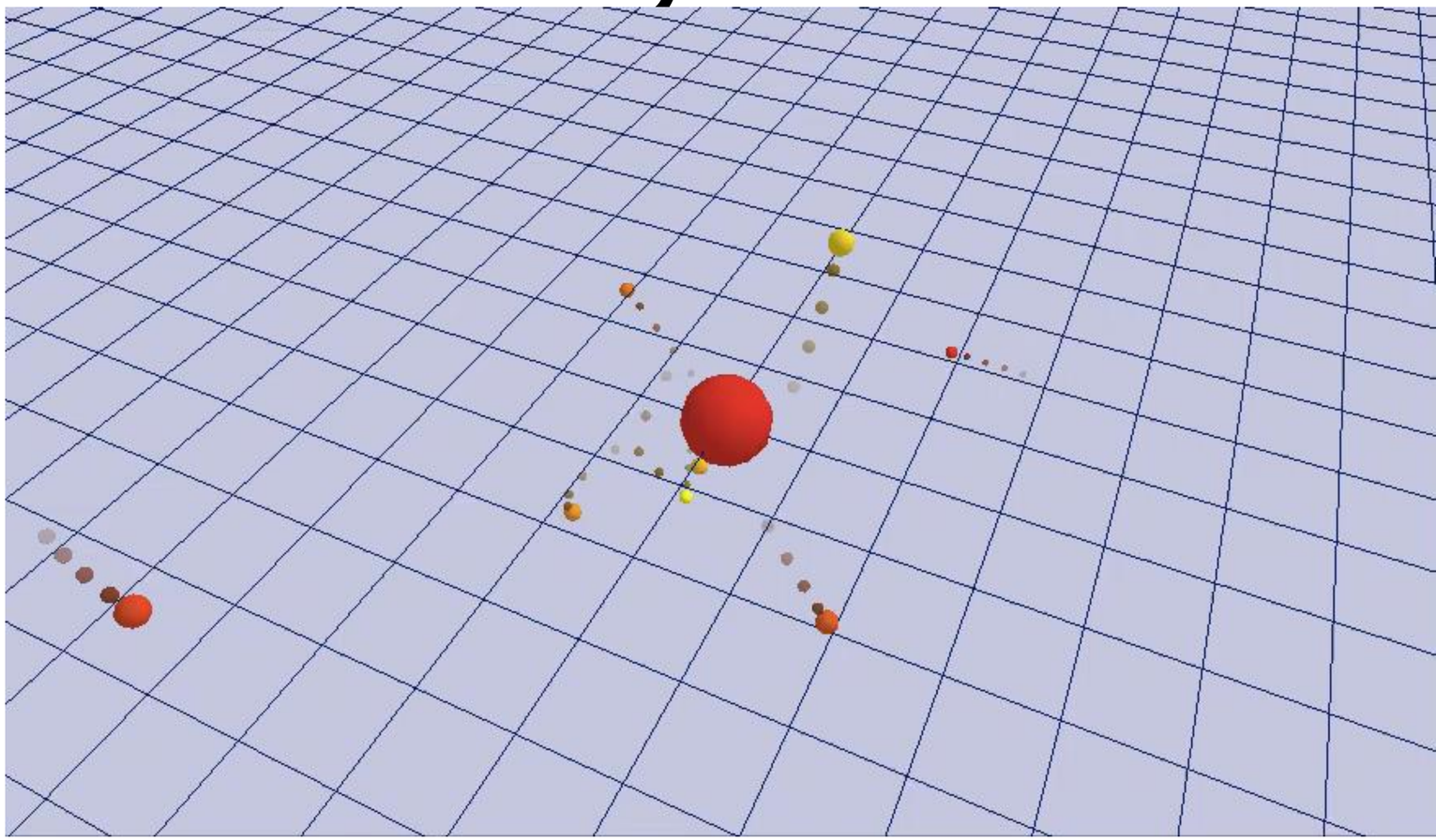


Force Models: Gravity Model

- (eq6) $\vec{f}_{ij} = \frac{G m_i m_j}{\|\vec{d}\|^3} \vec{d}$

- In a system without initial velocities the particles collapse

- When the velocity is perpendicular to the force, an orbit emerges

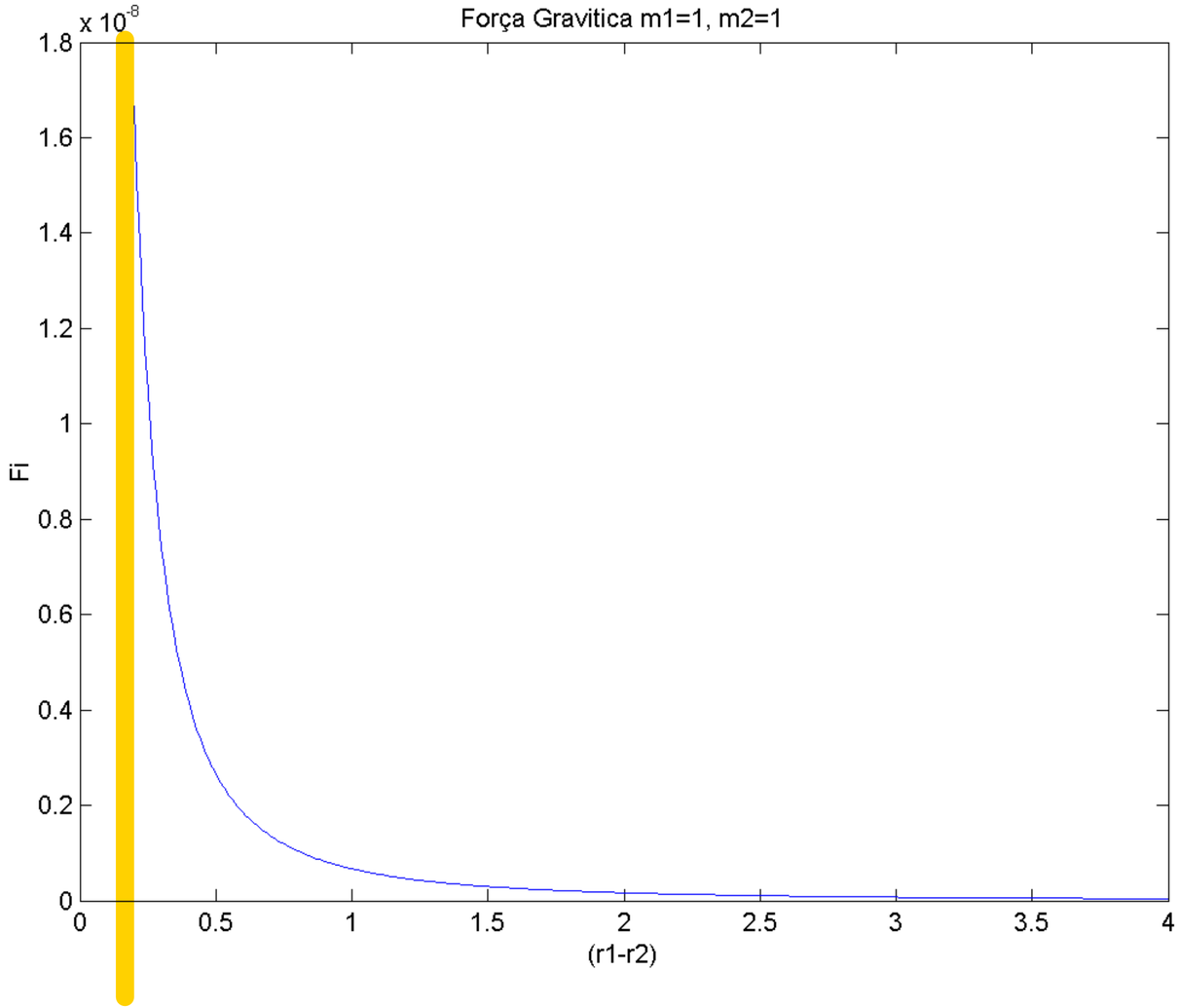


Force Models: Gravity Model

- (eq6) $\vec{f}_{ij} = \frac{G m_i m_j}{\|\vec{d}\|^3} \vec{d}$

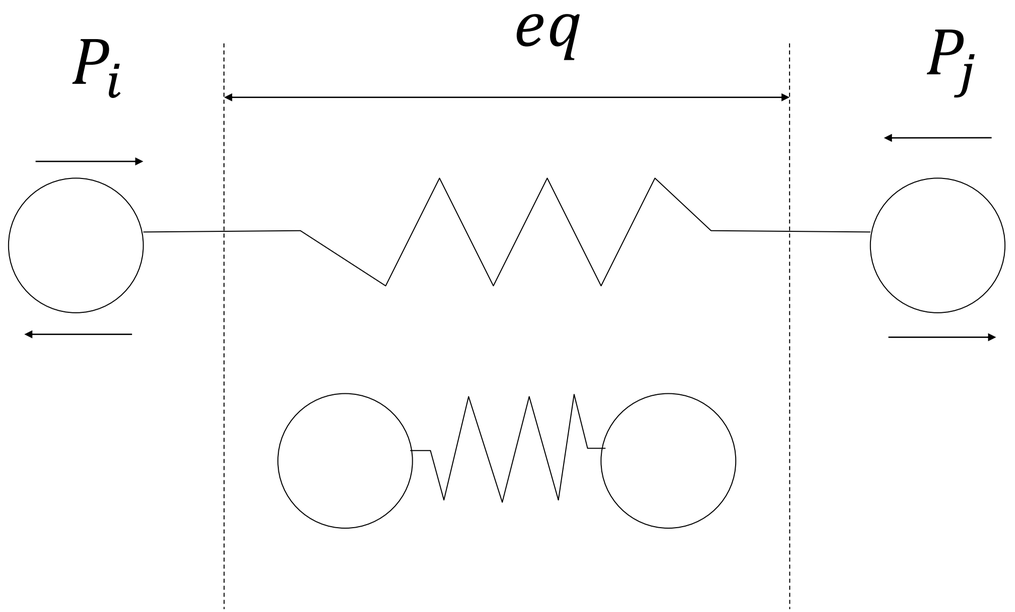
- As objects get infinitesimally close the force tends to infinity.

- In simulation there is sometimes a cutoff distance from which we consider that $d = 0$ and therefore there are no forces between the particles.



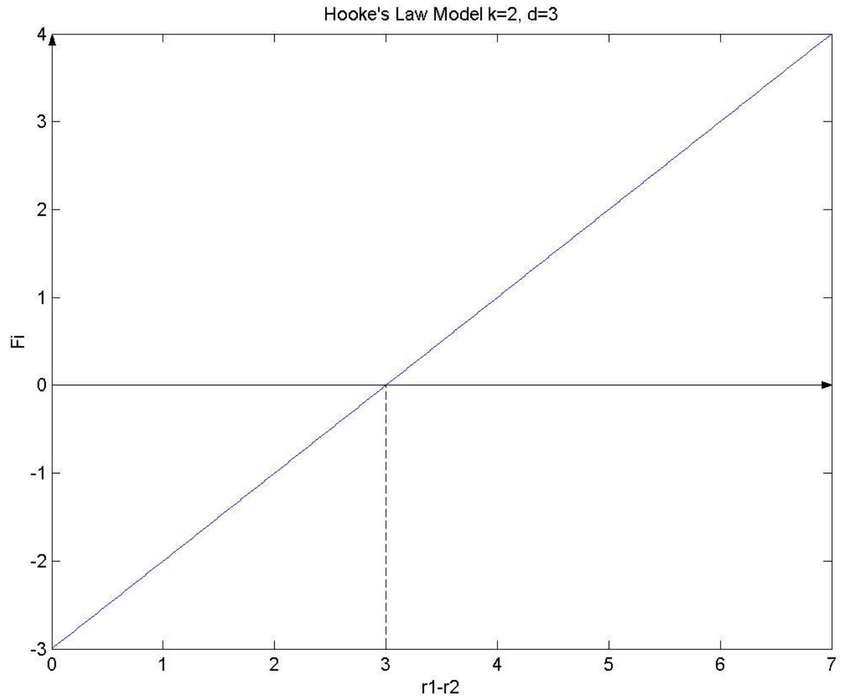
Force Models: Hooke Model

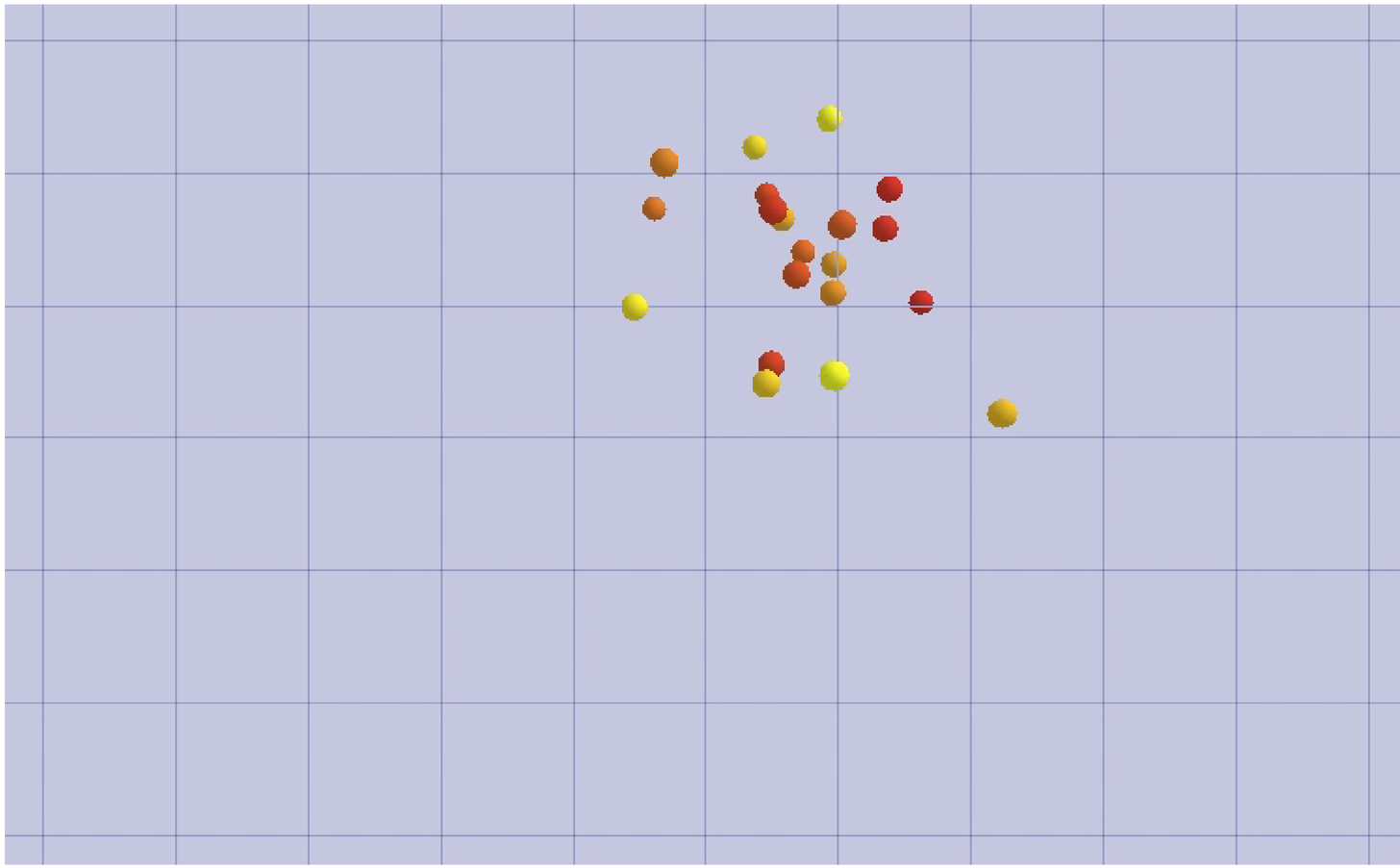
- Spring based model
- Particles are in a position of equilibrium eq with an elasticity of k.
- (eq7) $\|\vec{f}_{ij}\| = k(\|\vec{d}\| - eq)$
- Using (eq3) we get the vectorial form (eq8) $\vec{f}_{ij} = k\left(1 - \frac{\|eq\|}{\|\vec{d}\|}\right)\vec{d}$



Force Models: Hooke Model

- (eq8) $\vec{f}_{ij} = k\left(1 - \frac{\|eq\|}{\|\vec{d}\|}\right)\vec{d}$
- Particles stay together

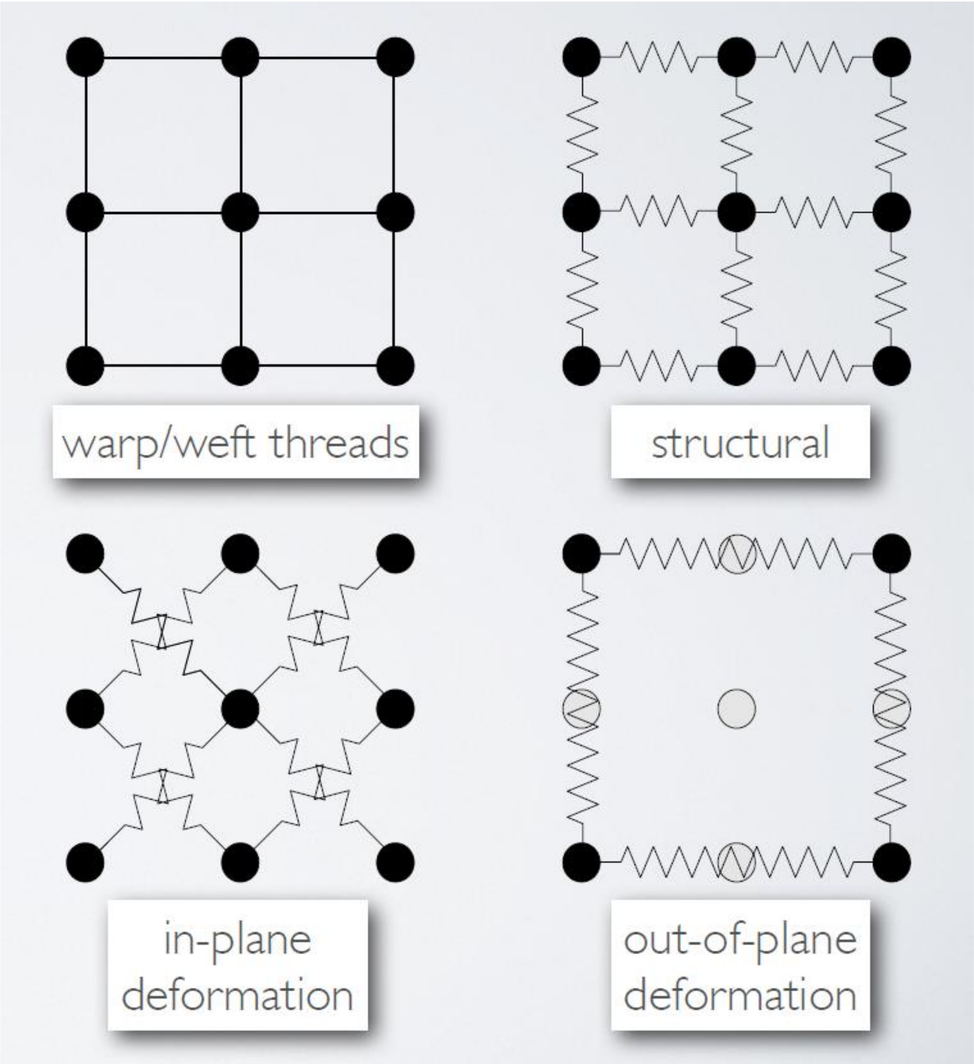




Fernando Birra
Rui Nóbrega

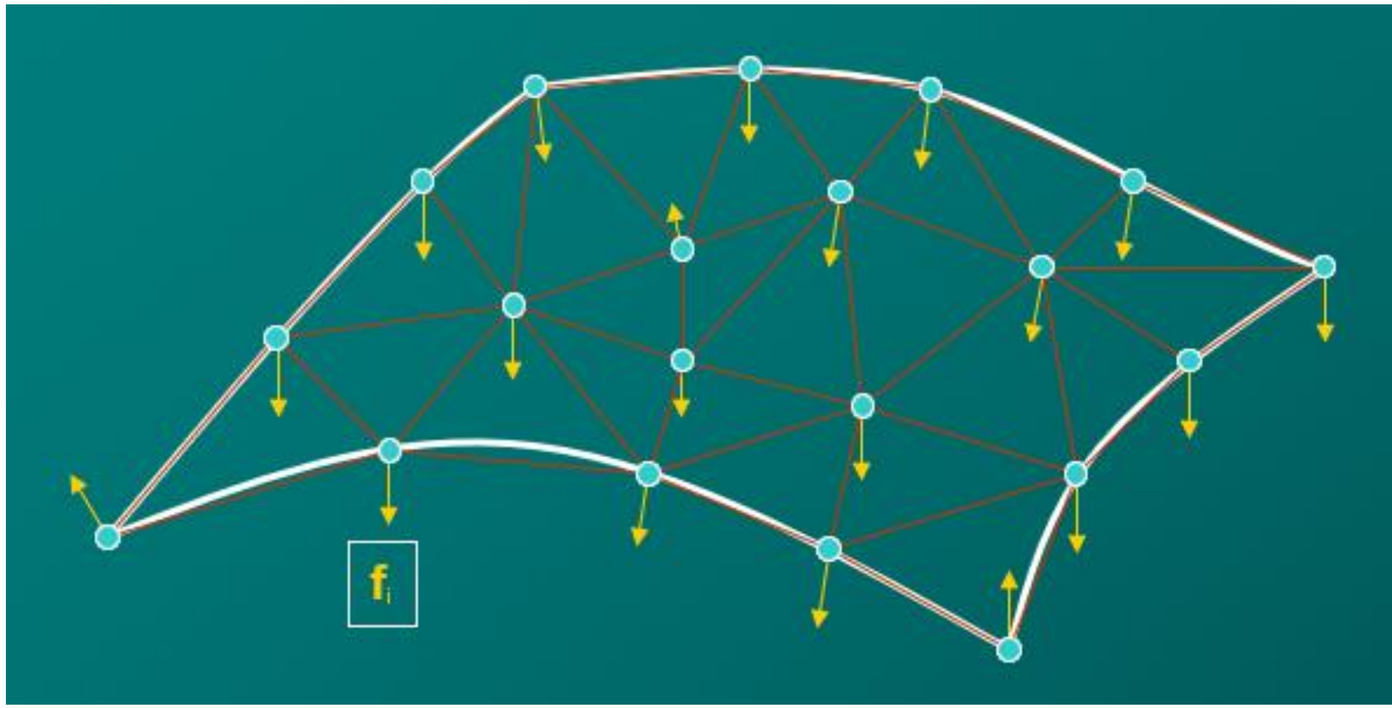
Force Models: Hooke Model

- Provot's Model
- Particles Masses can be thought as the intersection of threads
- Each interaction is captured through a set of springs
- Ideal for textiles, hair, viscosity

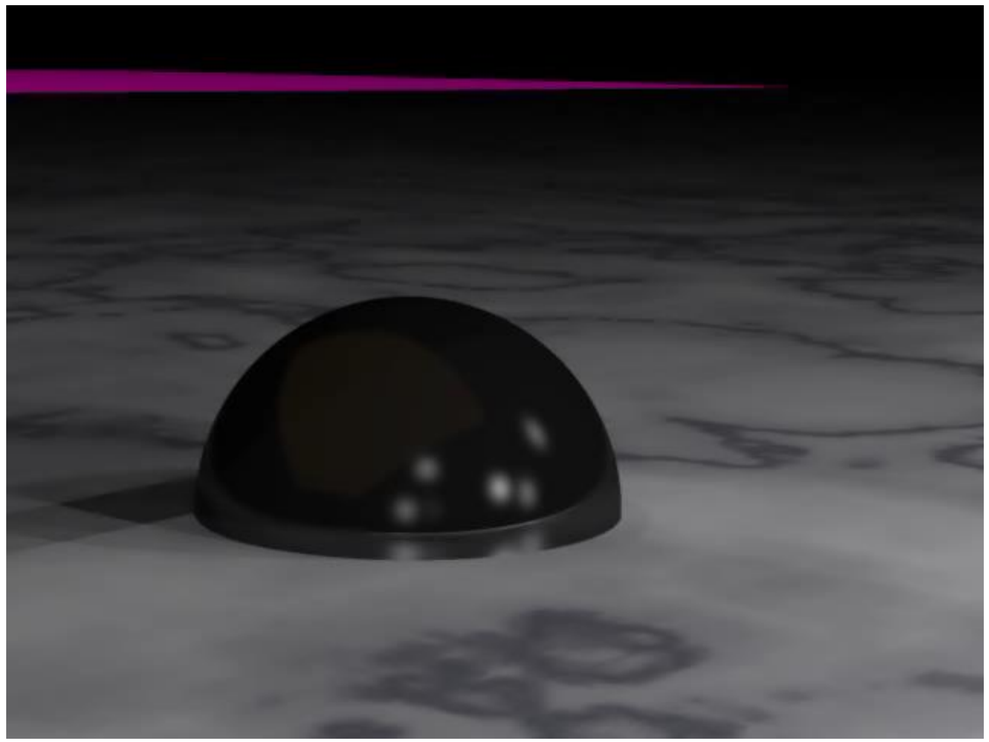


Force Models: Hooke Model

- Provot’s Model
- Particles Masses can be thought as the intersection of threads
- Each interaction is captured through a set of springs
- Ideal for textiles, hair, viscosity

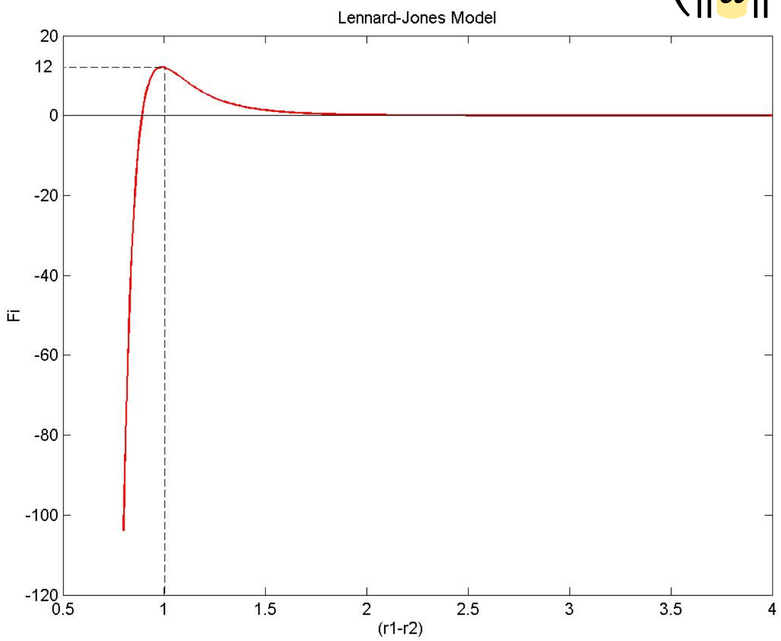


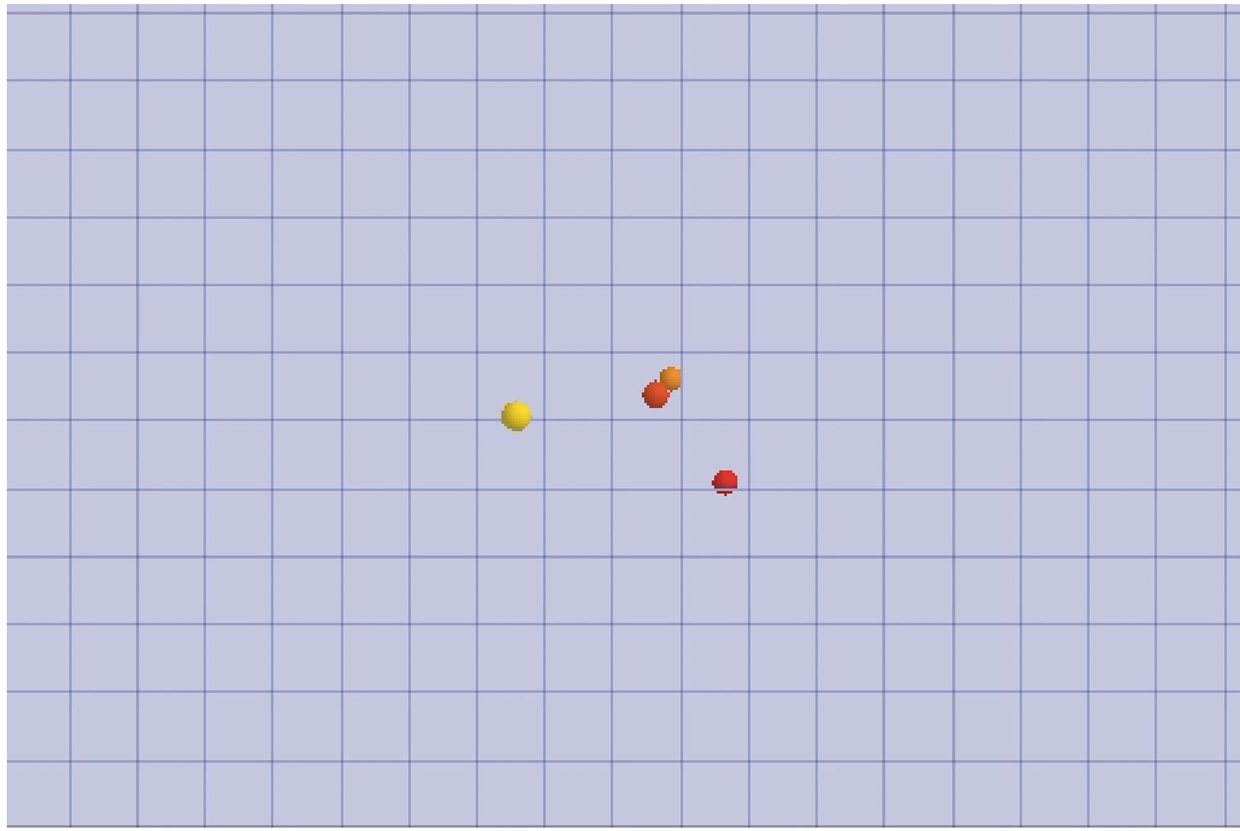
Force Models: Hooke Model



Force Models: Lennard-Jones

- Heavy realistic model
- Avoids collisions with repulsive force
- (eq9) $\vec{f}_{ij} = -12\left(\frac{1}{\|\vec{d}\|^{14}} - \frac{2}{\|\vec{d}\|^{8}}\right)\vec{d}$

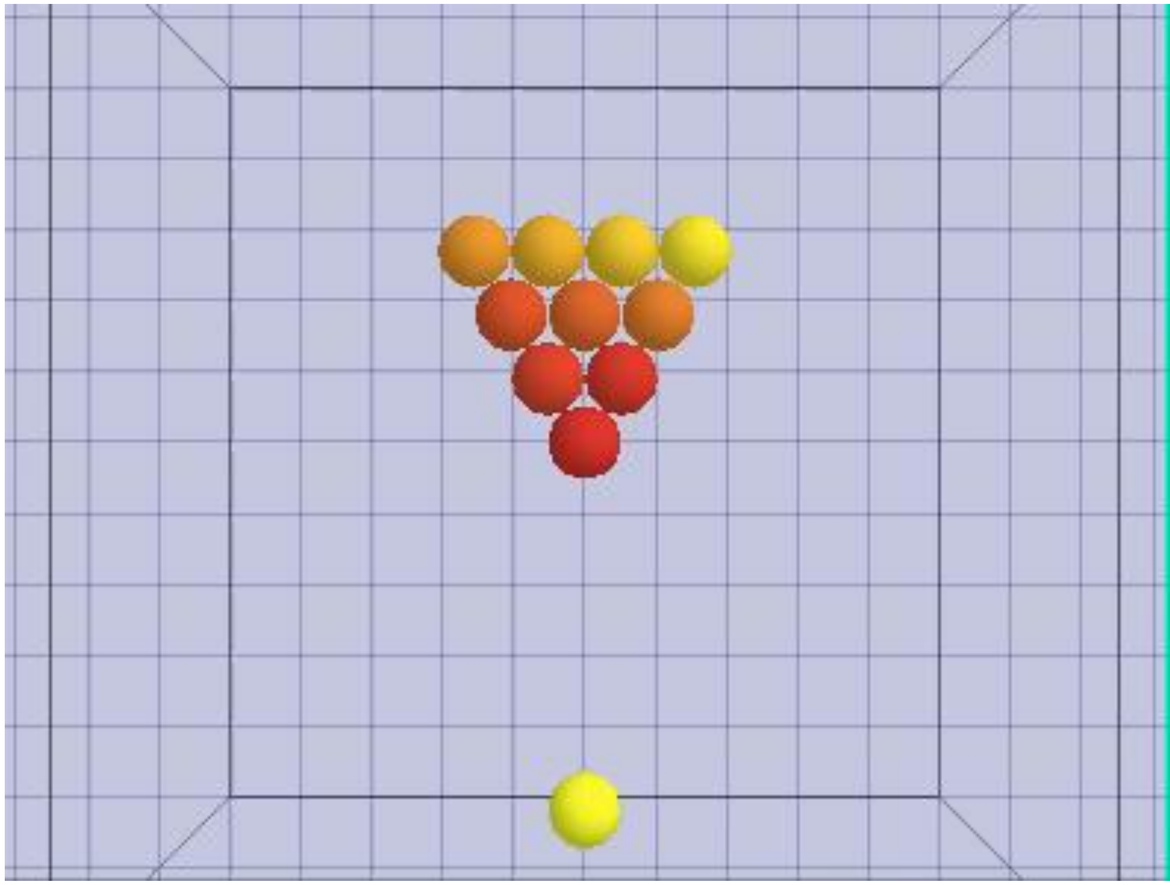




Fernando Birra
Rui Nóbrega

Force Models: Hard Spheres

- The system depends on all the kinetic energy that is inserted in the beginning of the simulation
- (eq10) $\vec{f}_{ij} = 0$
- It can sometimes be negative to simulate drag.



Particle Systems Parameters

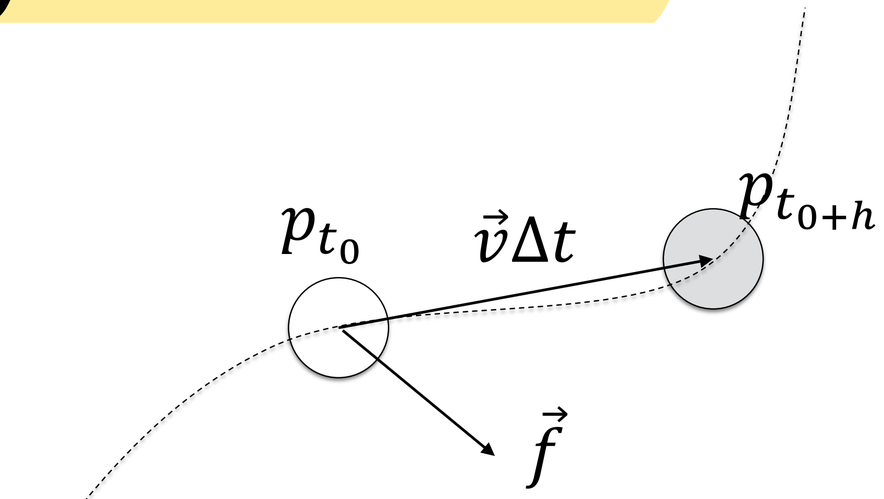
- In this particle systems simulation, there are several possible parameters:
  - **Force Model**: how should we calculate $\overrightarrow{f_{ij}}$?
    - Gravity model,
      Hooke's model,
      Lennard-Jones's model,
      Hard Spheres Model
  - **Interpolation Model**: how should we calculate $\Delta\overrightarrow{v_i}$ and $\Delta\overrightarrow{p_i}$?
    - Euler,
      Verlet
  - **Collision Model**: how should we calculate collisions?
    - Hard Spheres

Particle Systems Parameters

- In this particle systems simulation, there are several possible parameters:
  - **Force Model**: how should we calculate $\overrightarrow{f_{ij}}$?
    - Gravity model,
      Hooke's model,
      Lennard-Jones's model,
      Hard Spheres Model
  - **Interpolation Model**: how should we calculate $\Delta\overrightarrow{v_i}$ and $\Delta\overrightarrow{p_i}$?
    - Euler,
      Verlet
  - **Collision Model**: how should we calculate collisions?
    - Hard Spheres

Interpolation: Taylor series

- Given a certain applied force what is the new position velocity v and the new position p?



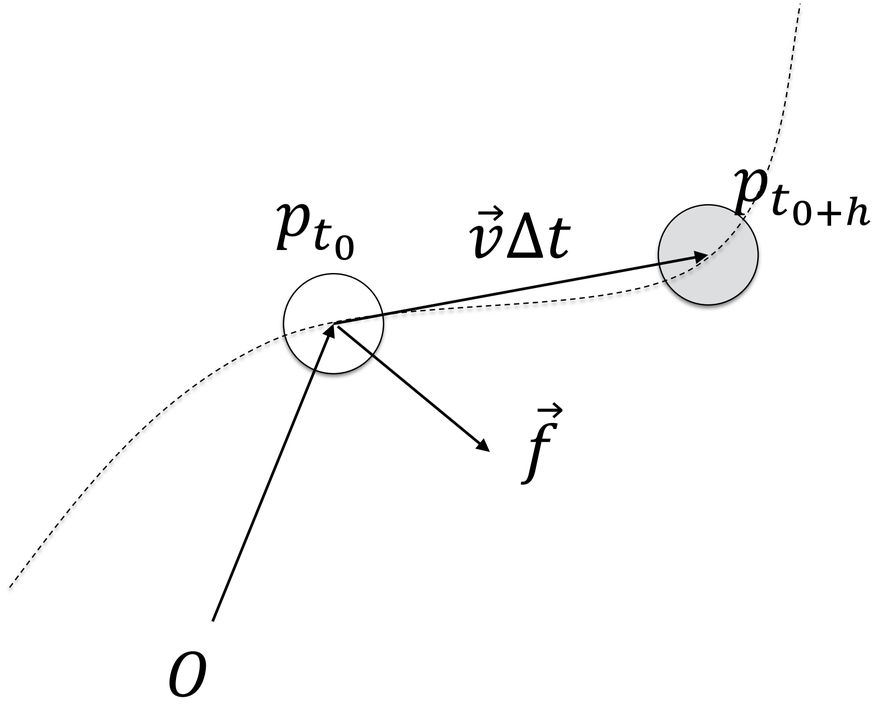
- Taylor's series expansion (eq11):

$$f(x_0 + h) = f(x_0) + h\frac{df}{dx}(x_0) + \frac{h^2}{2!}\frac{d^2f}{dx^2}(x_0) + \cdots + \frac{h^n}{2!}\frac{d^nf}{dx^n}(x_0)$$

- $f(x_0 + h) = f(x_0) + h\frac{df}{dx}(x_0) + \varepsilon$, where ε is the error of the simulation (example for Euler method).

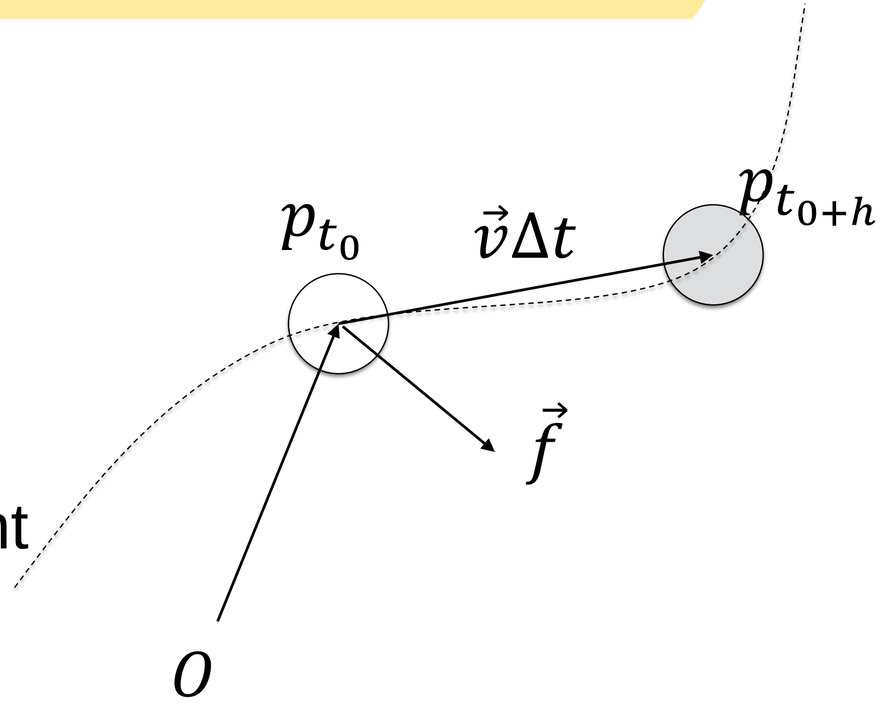
Interpolation: Euler Method

- The Euler model is a fast simplified method using the first two terms of the Taylor series (eq11) $(h = \Delta t)$
- (eq12) $\vec{f} = m\vec{a}, \vec{a} = \frac{\partial \vec{v}}{\partial t}$
- (eq13) $\overrightarrow{\Delta v_i} = \frac{\Delta t}{m_i}\overrightarrow{f_i}$ is the increment in speed
- (eq14) $\Delta\overrightarrow{p_i} = \overrightarrow{v_i}\Delta t$



Interpolation: Verlet Method

- The Verlet model is a more realistic method using three terms of the Taylor series (eq11)
- Accumulates less errors for the same amount of steps as the Euler method
- (eq15) $\Delta\overrightarrow{p_i} = \overrightarrow{v_i}\Delta t + \frac{1}{2}\overrightarrow{a_i}(\Delta t)^2$
- (eq16) $\Delta\vec{v} = \frac{\Delta t}{2}(2\overrightarrow{a_i} + \Delta a)$



Particle Systems Parameters

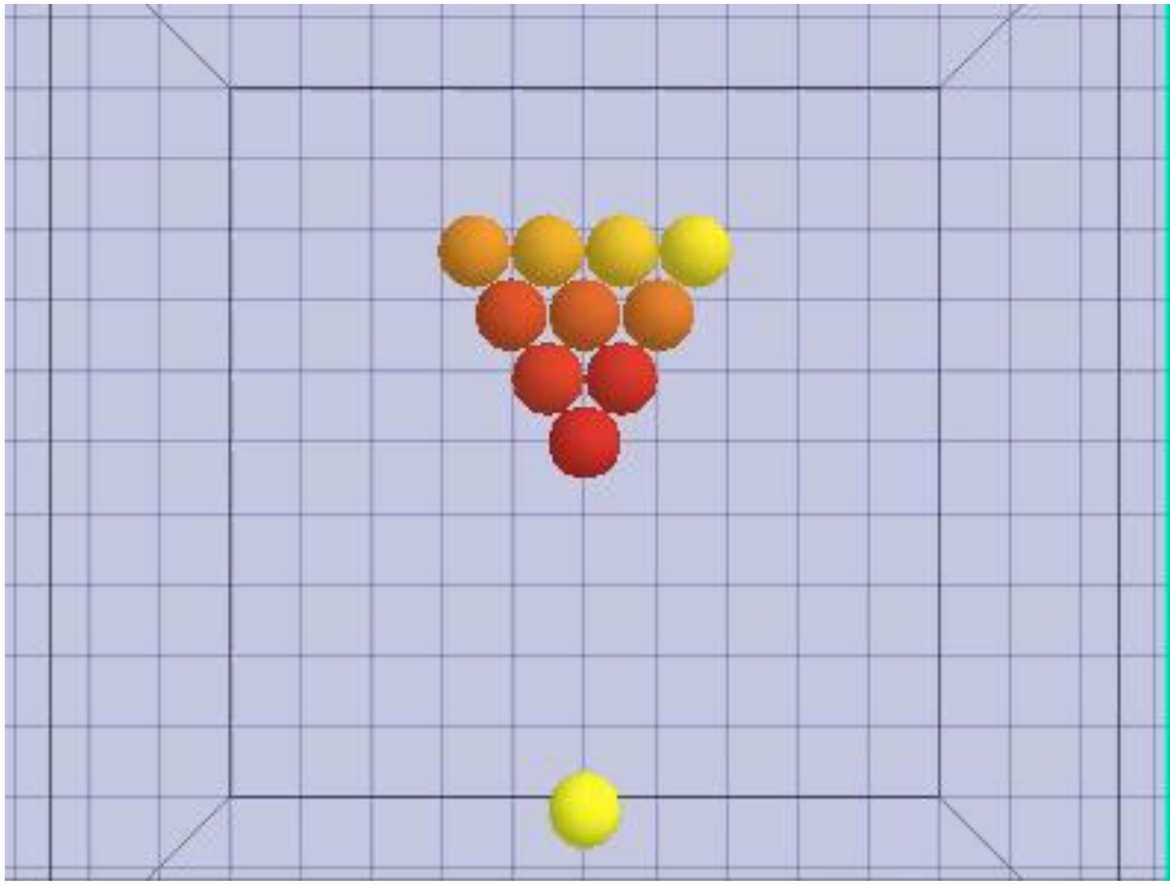
- In this particle systems simulation, there are several possible parameters:
  - **Force Model**: how should we calculate $\overrightarrow{f_{ij}}$?
    - Gravity model,
      Hooke's model,
      Lennard-Jones's model,
      Hard Spheres Model
  - **Interpolation Model**: how should we calculate $\Delta\overrightarrow{v_i}$ and $\Delta\overrightarrow{p_i}$?
    - Euler,
      Verlet
  - **Collision Model**: how should we calculate collisions?
    - Hard Spheres

Particle Systems Parameters

- In this particle systems simulation, there are several possible parameters:
  - **Force Model**: how should we calculate $\overrightarrow{f_{ij}}$?
    - Gravity model,
      Hooke's model,
      Lennard-Jones's model,
      Hard Spheres Model
  - **Interpolation Model**: how should we calculate $\Delta\overrightarrow{v_i}$ and $\Delta\overrightarrow{p_i}$?
    - Euler,
      Verlet
  - **Collision Model**: how should we calculate collisions?
    - Hard Spheres

Collision Model: Hard Spheres

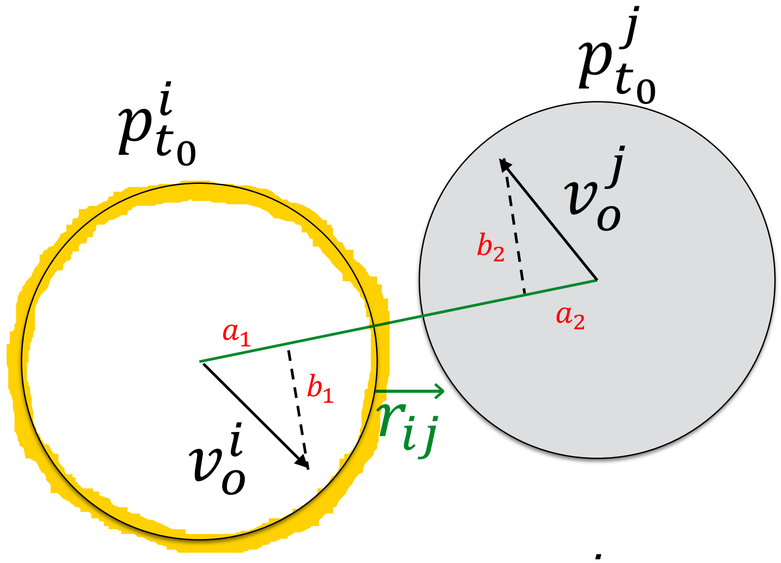
- Every particle has a diameter σ
- At any time, particles maintain the linear momentum using the linear momentum conservation law
- (eq17) $m_i \overrightarrow{v_o^i} + m_j \overrightarrow{v_o^j} = m_i \overrightarrow{v_h^i} + m_j \overrightarrow{v_h^j}$
- Pool table balls effect



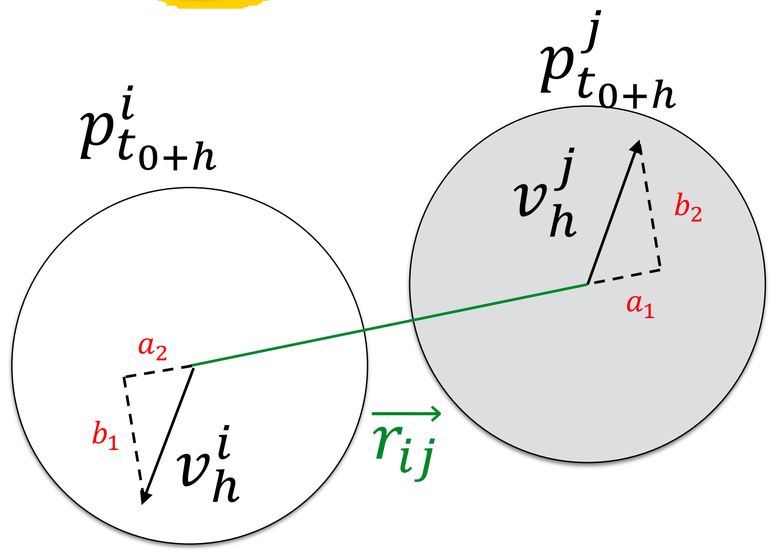
Collision Model: Hard Spheres

- Considering a collision that happens between a give time interval h.
- After a collision the velocity components that coincide with the collision axis $\overrightarrow{r_{ij}}$ (a1 and a2) swap.
- The other components maintain its value (b1 and b2).

Before collision



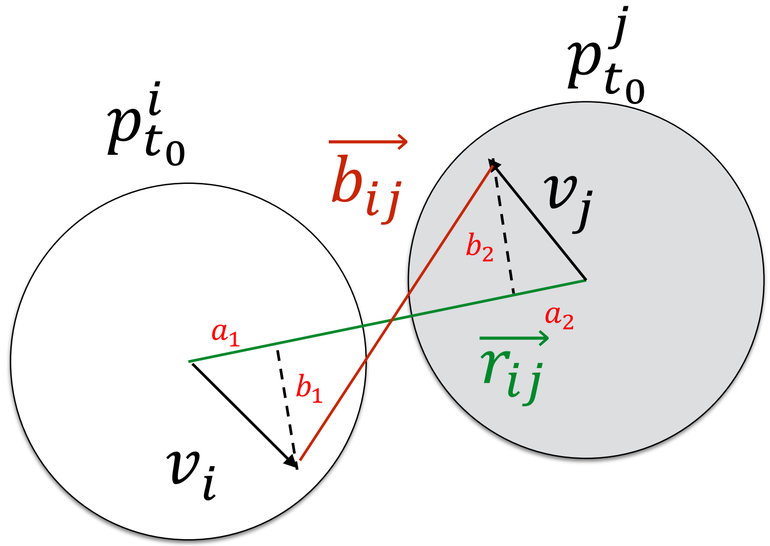
After collision



Collision Model: Hard Spheres

Detecting collisions:

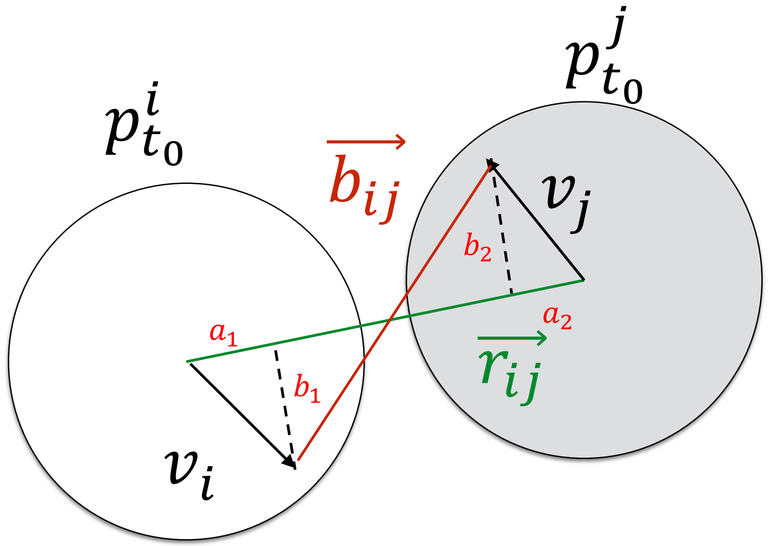
- Are they getting closer?
  - (eq18) $\overrightarrow{v_{ij}} = \overrightarrow{v_j} - \overrightarrow{v_i}$
  - (eq19) $\overrightarrow{r_{ij}} \cdot \overrightarrow{v_{ij}} < 0$
- Is it inside the radius of the other?
  - Given the current velocities will (eq20) $\left\|\overrightarrow{b_{ij}}\right\| < \sigma$?



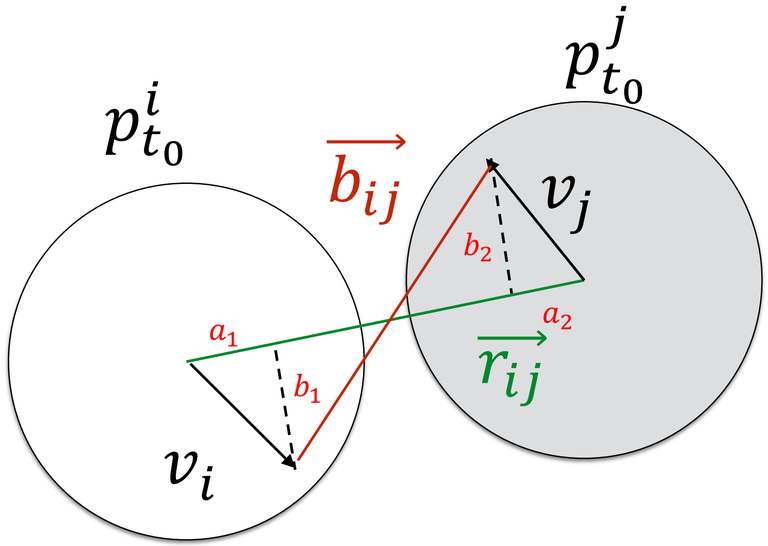
Collision Model: Hard Spheres

Detecting collisions:

- Is it inside the radius of the other?
  - Given the current velocities will the two spheres intersect?
    (eq20) $\|\overrightarrow{b_{ij}}\| < \sigma$
  - (eq21) $\|\overrightarrow{b_{ij}}\|^2 = \|\overrightarrow{r_{ij}}\|^2 - \left(\frac{\overrightarrow{r_{ij}} \cdot \overrightarrow{v_{ij}}}{\|\overrightarrow{v_{ij}}\|}\right)^2$



Collision Model: Hard Spheres



Detecting collisions:

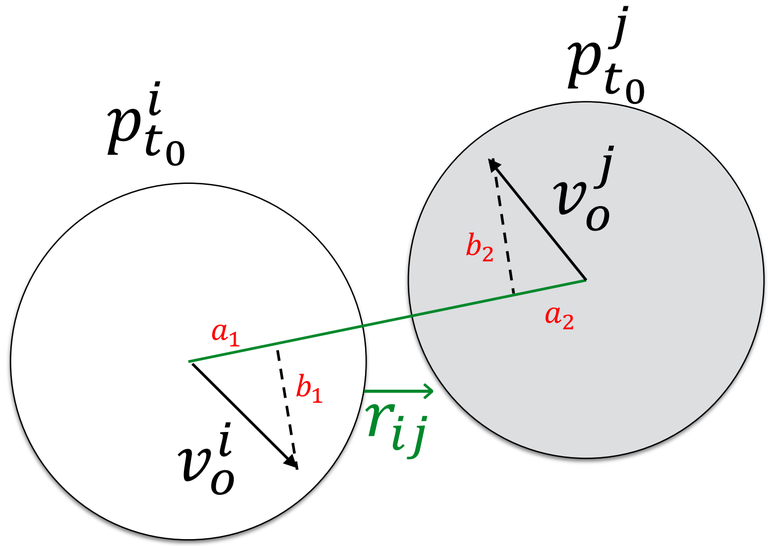
- When will the collision be?
  - (eq22) $t = -\frac{1}{\|\overrightarrow{v_{ij}}\|}\left(\frac{\overrightarrow{r_{ij}}\cdot\overrightarrow{v_{ij}}}{\|\overrightarrow{v_{ij}}\|} + \left(\sigma^2 - \|\overrightarrow{b_{ij}}\|^2\right)^{1/2}\right)$

Collision Model: Hard Spheres

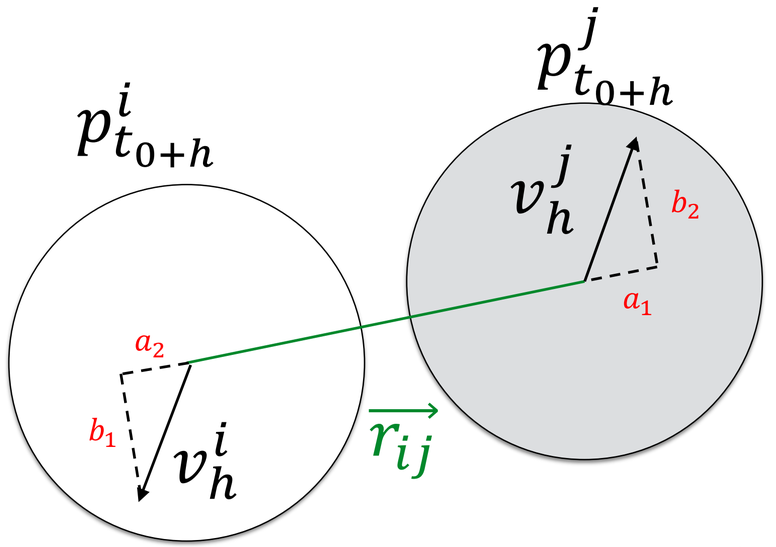
Updating velocities:

- After a collision, the velocity components that coincide with the collision axis $\overrightarrow{r_{ij}}$ (a1 and a2) swap.
- The other components maintain its value (b1 and b2).
- (eq23) $\overrightarrow{v_h^i} = \overrightarrow{v_o^i} + \Delta\overrightarrow{v_o^i}, \qquad \overrightarrow{v_h^j} = \overrightarrow{v_o^j} + \Delta\overrightarrow{v_o^i}$
- (eq24) $\Delta\overrightarrow{v_o^i} = -\frac{(\overrightarrow{r_{ij}}\cdot\overrightarrow{v_{ij}})\overrightarrow{r_{ij}}}{\|\overrightarrow{r_{ij}}\|^2}$

Before collision

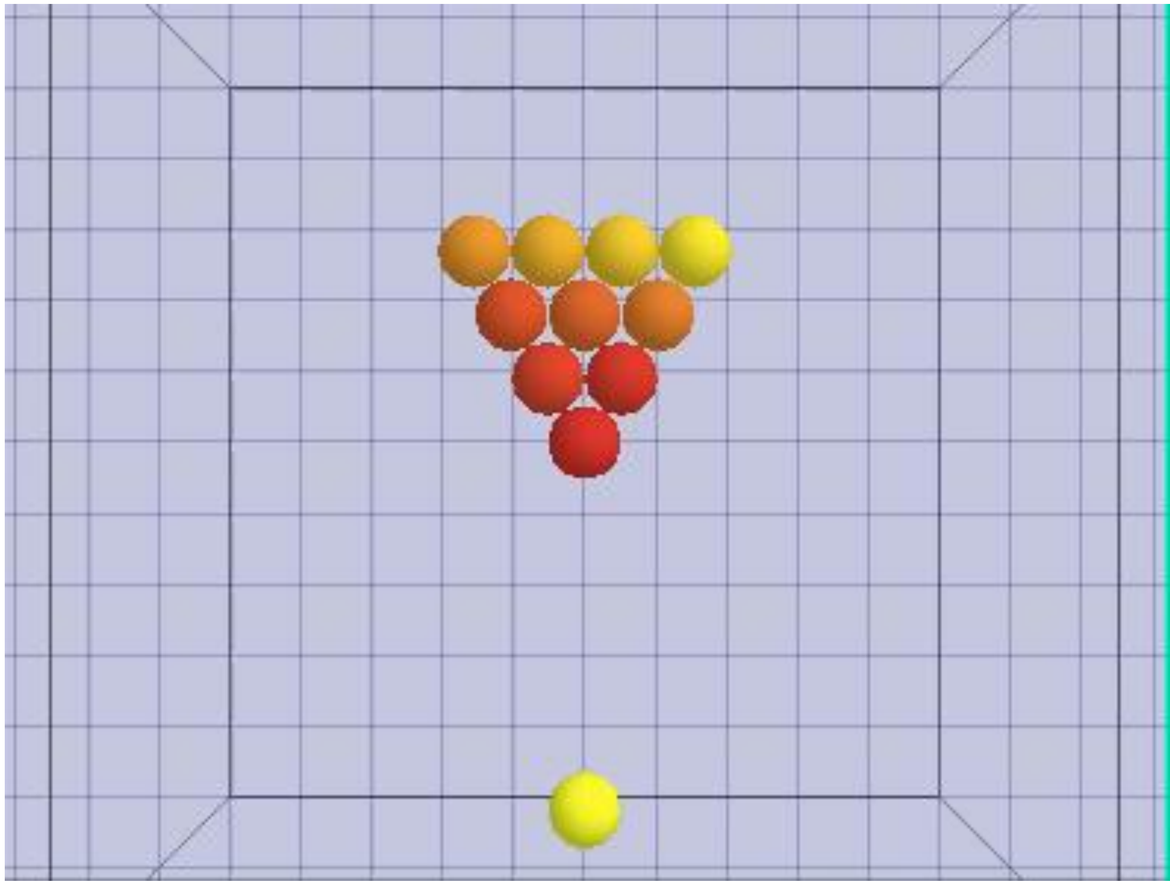


After collision



Force Models: Hard Spheres

- The system depends on all the kinetic energy that is inserted in the beginning of the simulation
- (eq10) $\vec{f}_{ij} = 0$
- Great for non-elastic collisions (pool balls).
- Can be used in conjunction to other force models.
- Limited to spheres.



Particle Systems Summary

- In this particle systems simulation, there are several possible parameters:
  - **Force Model**: how should we calculate $\overrightarrow{f_{ij}}$?
  - **Interpolation Model**: how should we calculate $\Delta\overrightarrow{v_i}$ and $\Delta\overrightarrow{p_i}$?
  - **Collision Model**: how should we calculate collisions?

- Gravity model,
  Hooke's model,
  Lennard-Jones's model,
  Hard Spheres Model
- Euler,
  Verlet
- Hard Spheres