



Inside LiquidFun



Kentaro Suto
Senior Software Engineer
Google

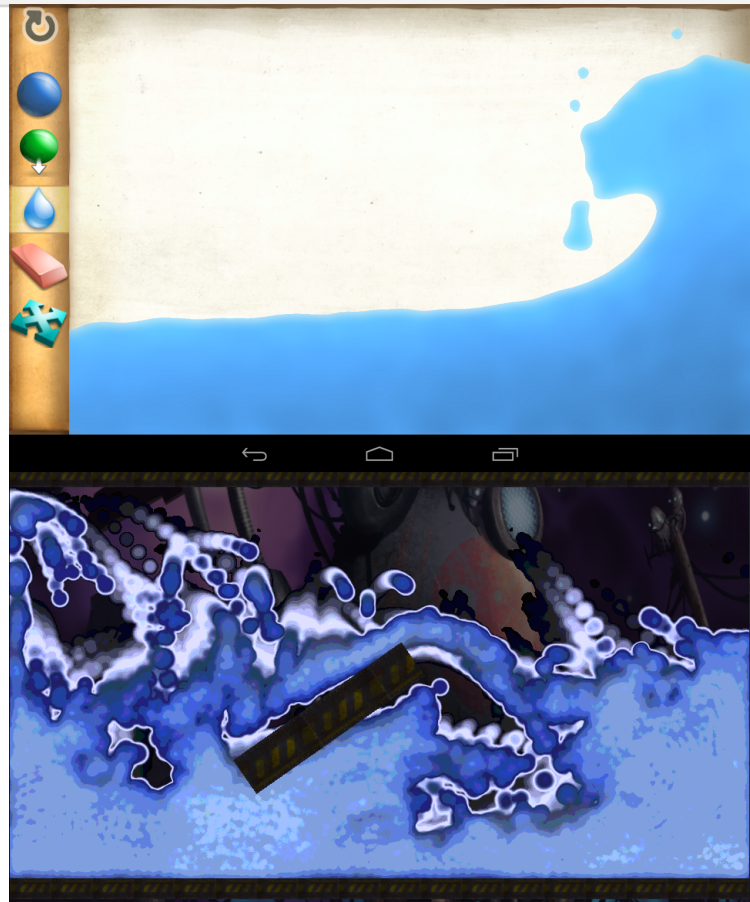
Contents

Overview of LiquidFun

What is a particle simulation?

Data structures

Simulation Algorithm



Overview of LiquidFun

- 2D rigid body and fluid simulation library
- Extension of Box2D
- Written in C++
- Runs on Android, iOS, Windows, OS X, Linux
- Distinguishing feature: particle simulation

What is particle simulation?

In particle simulation...

- objects are composed of particles

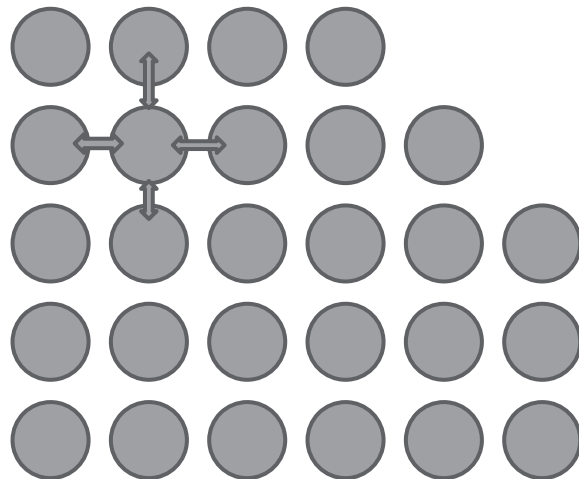
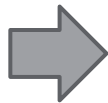
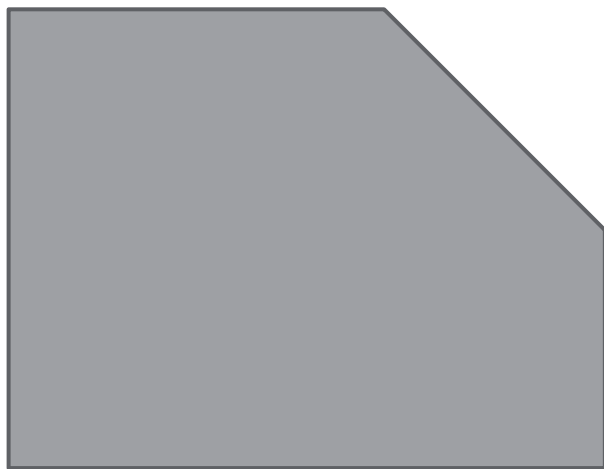
- physics is described by interactions between particles

Particle simulation works well with...

- Fluids (liquids and gasses)

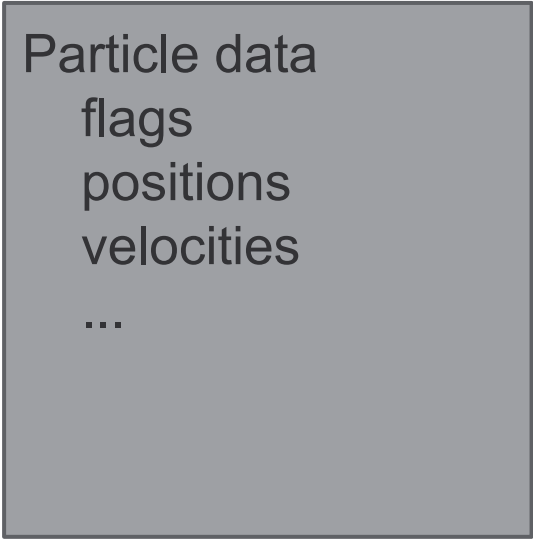
- Deformable objects

- Interactions of different materials



Particle simulation

Data structure

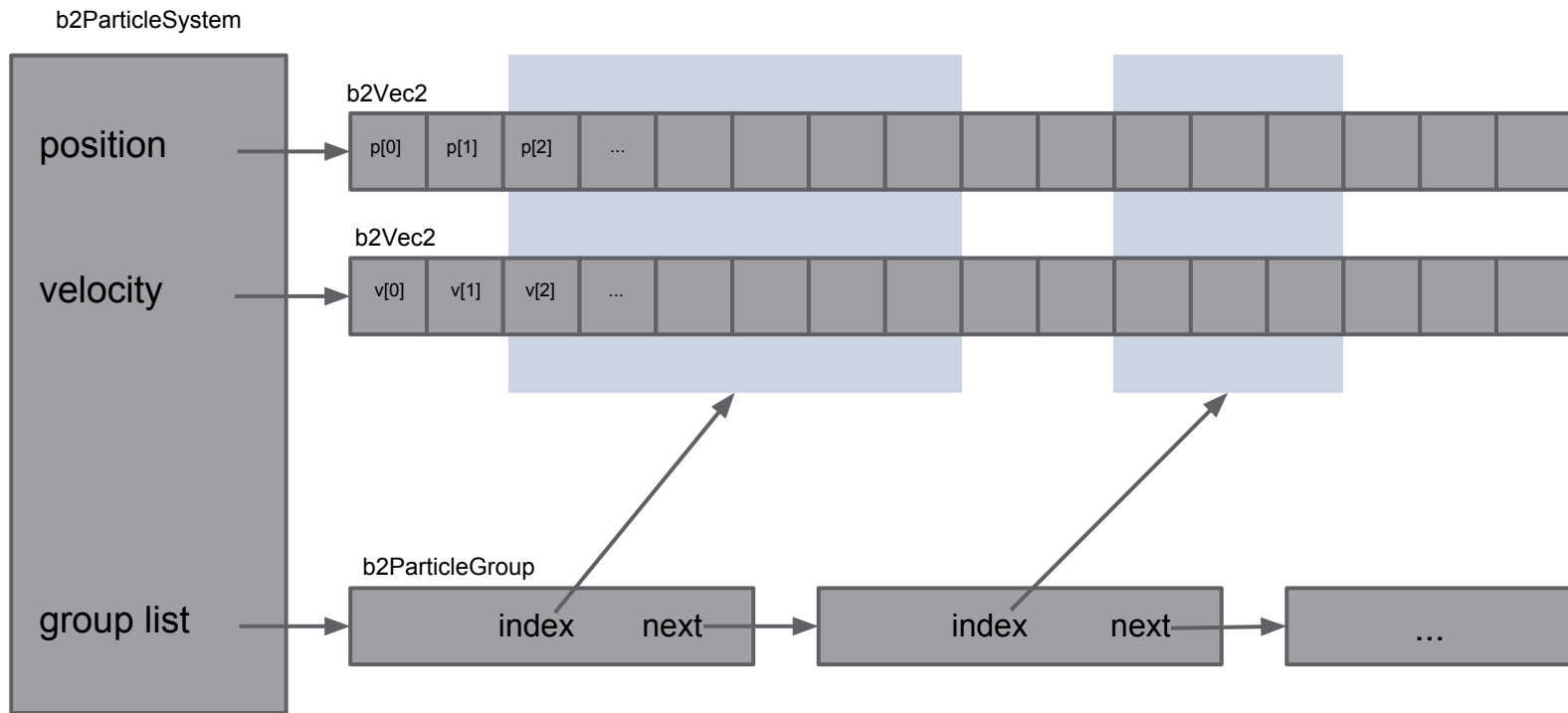


Particle data
flags
positions
velocities
...

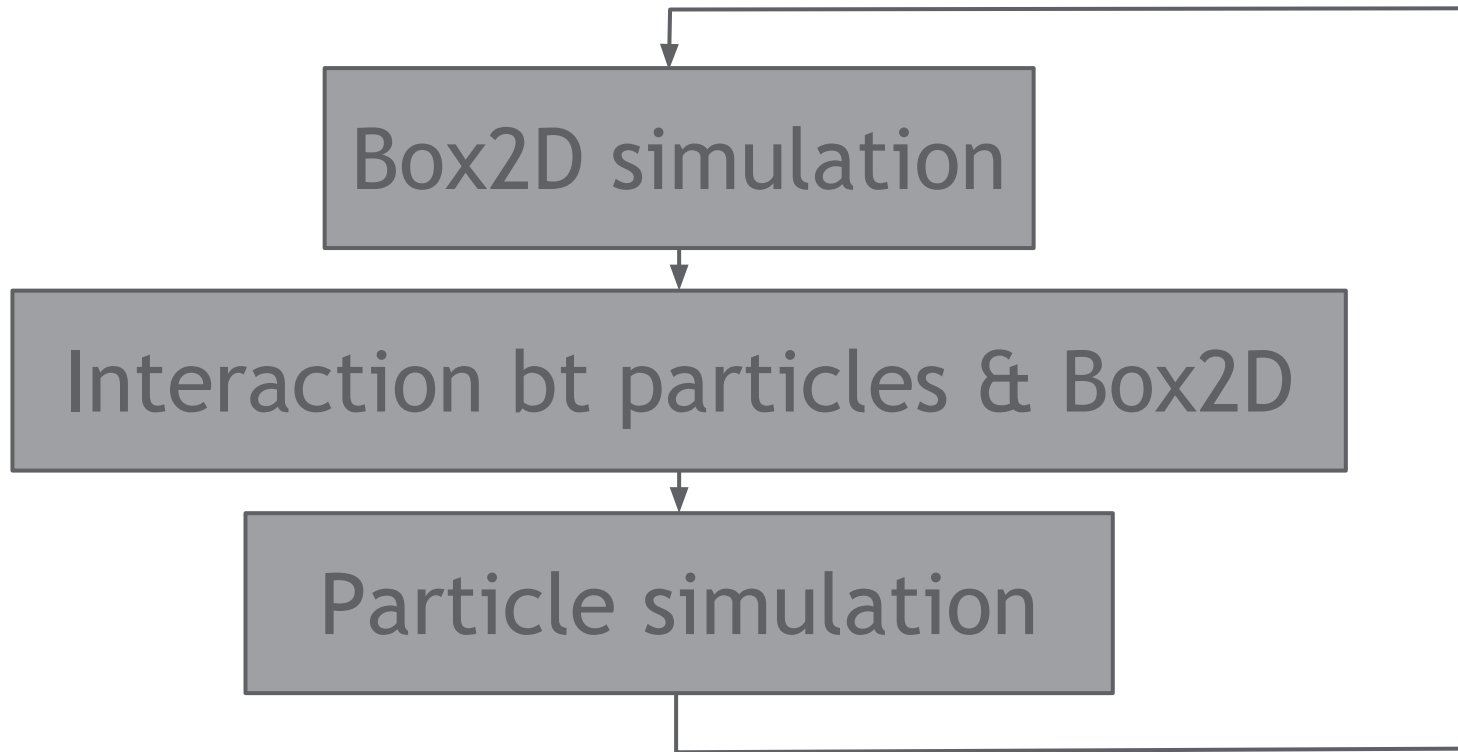


Particle group
flags
first index
last index
...

Data structure



Simulation Algorithm



Legend

$p[i]$: position of particle i

$v[i]$: velocity of particle i

Δt : simulation time step

R : particle radius

D : particle diameter = $2 * R$



Particle simulation

Collision detection

Apply pressure

Apply additional forces (viscous, etc.)

Restrict particle velocity (rigid, wall, etc.)

Move particles



Collision detection

Find pairs of particles closer than the particle diameter

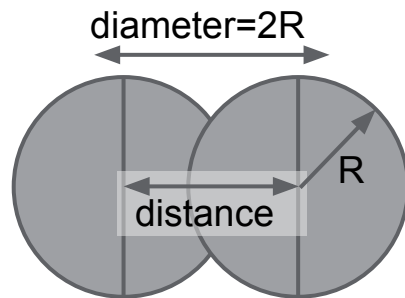
Record the following values:

i, j : indices of colliding particles

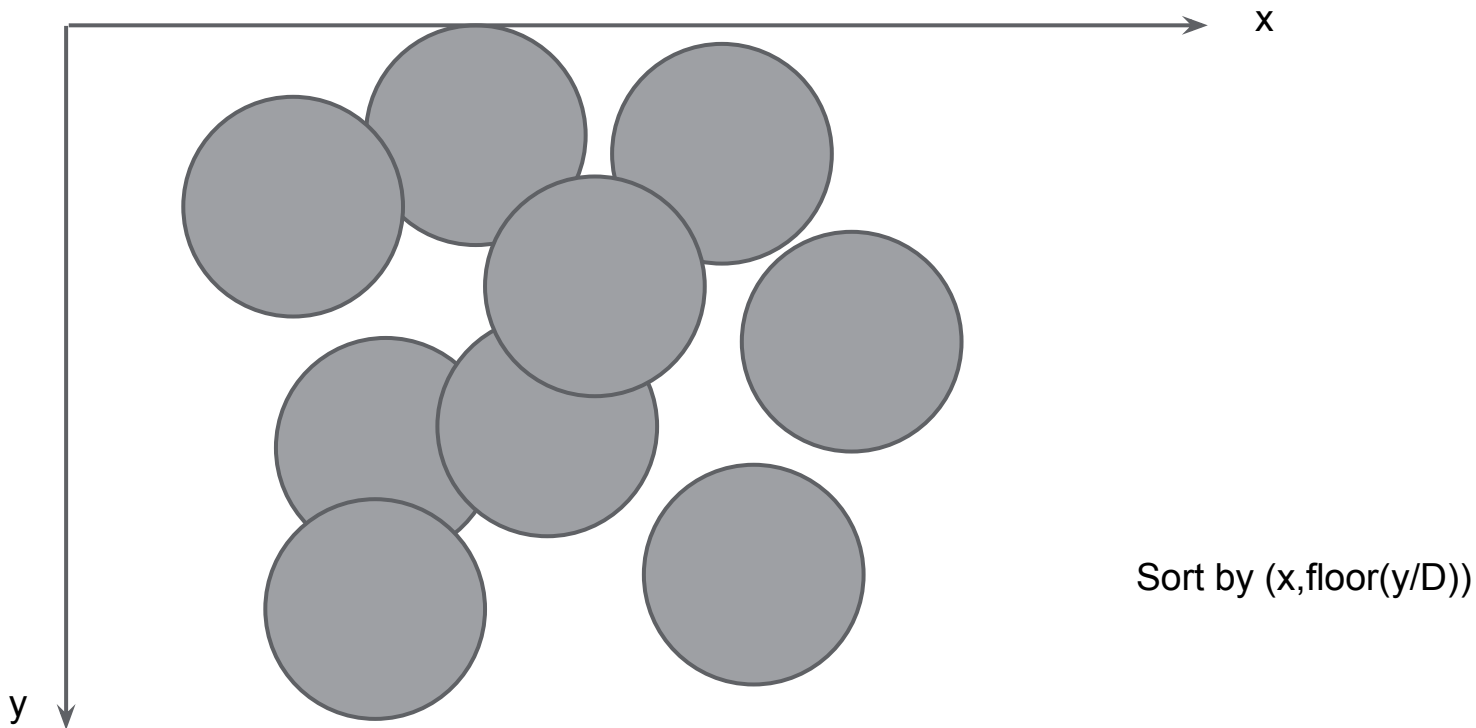
$n[i, j]$: normalized relative position

$w[i, j]$: calculated as $1 - \text{distance} / \text{diameter}$

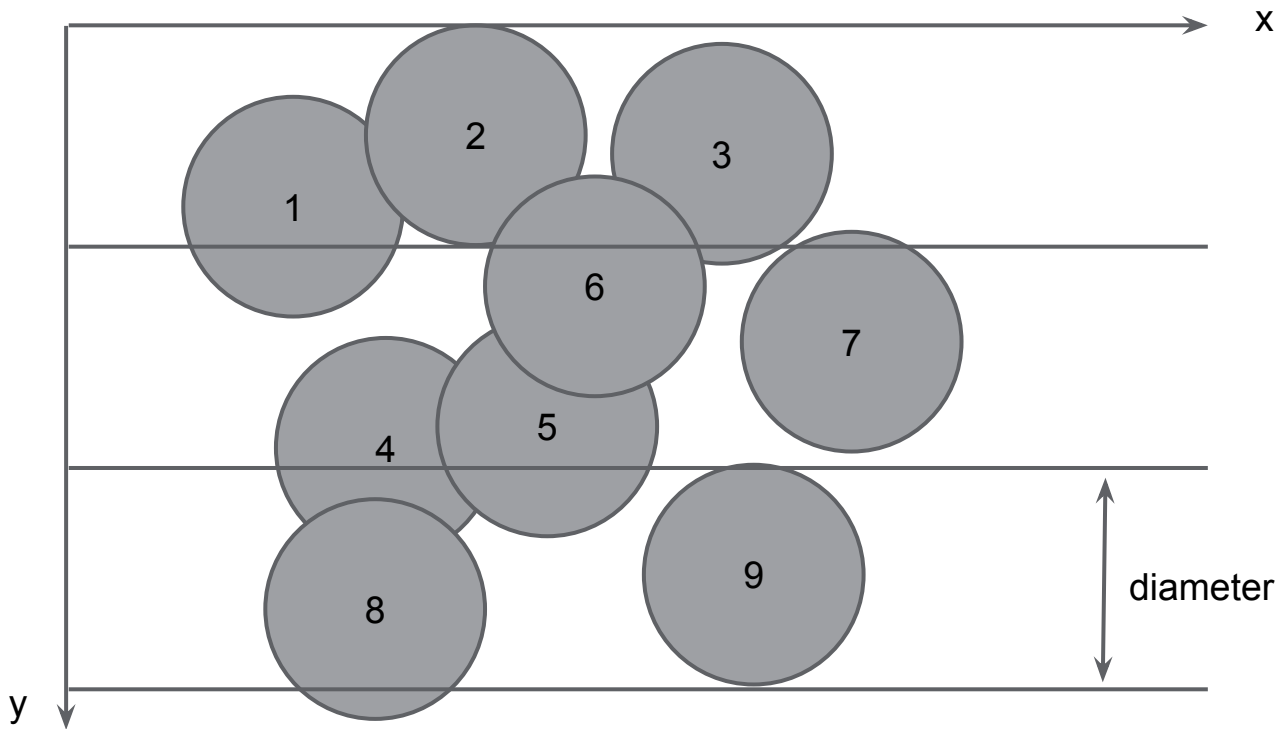
0 if barely contacting, 1 if overlapping



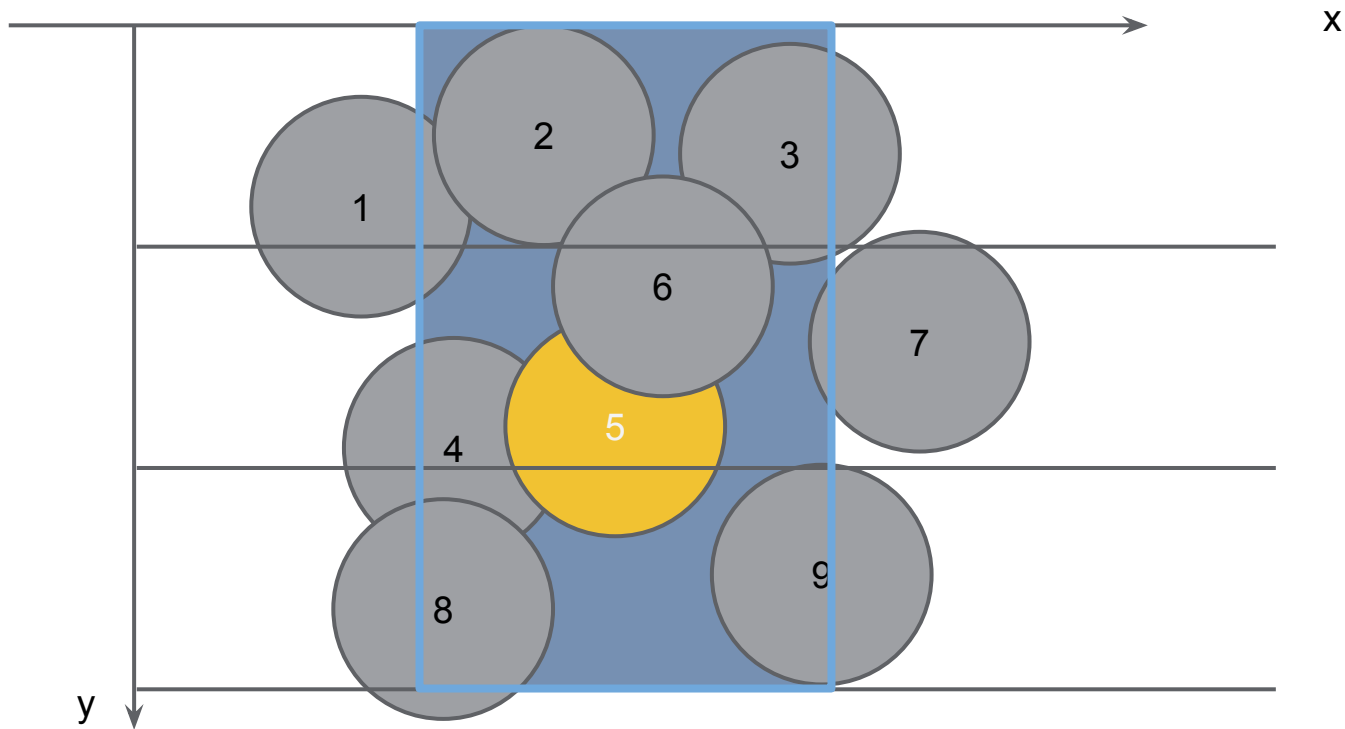
Collision detection - details(1)



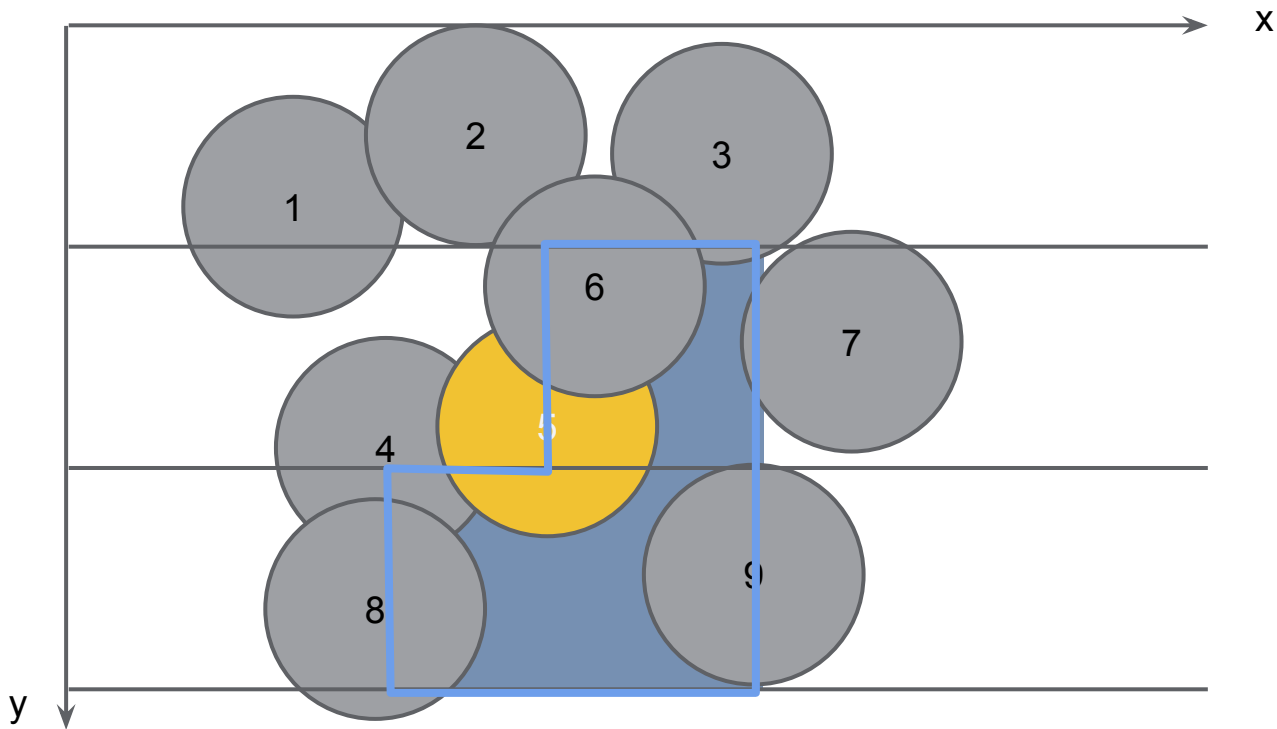
Collision detection - details(2)



Collision detection - details(3)



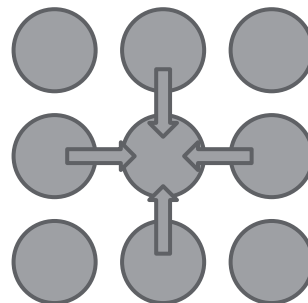
Collision detection - details(4)



Apply pressure(1)

Sum up the weight of contacts for each particle

$$w[i] = \text{sum}(w[i,j], j)$$



Apply pressure(2)

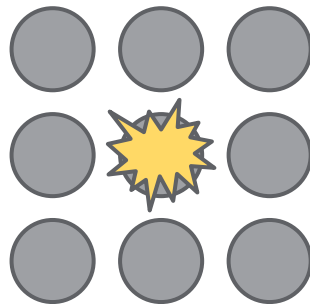
Calculate the pressure of each particle using the weight sum

$$h[i] = \max(0, \eta * (w[i] - w_0))$$

$h[i]$: pressure of particle i

η : constant coefficient

w_0 : constant average weight

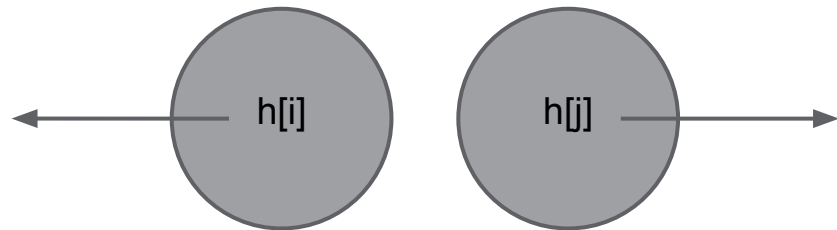


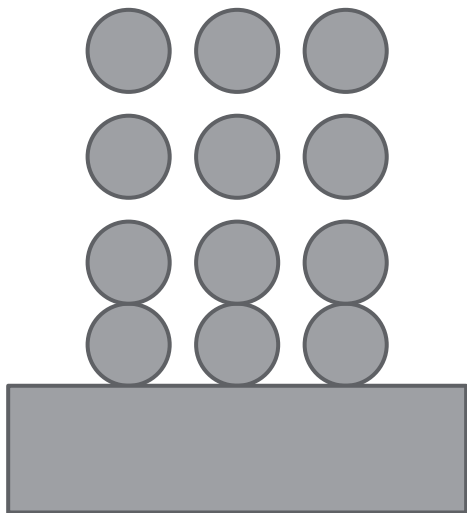
Apply pressure(3)

Apply a repulsive force to each contacting particle according to pressure

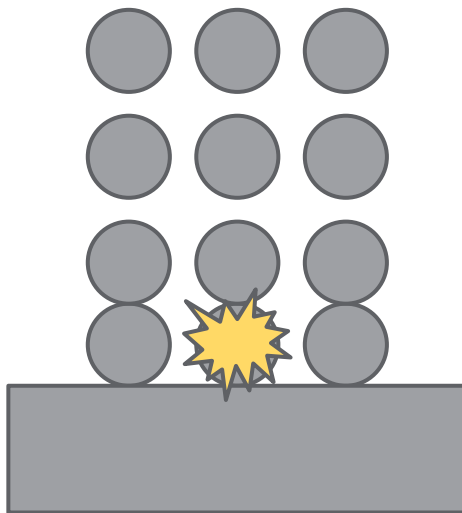
$$v[i] \leftarrow v[i] + \Delta t * a * (h[i] + h[j]) * w[i,j] * n[i,j]$$

a: constant coefficient

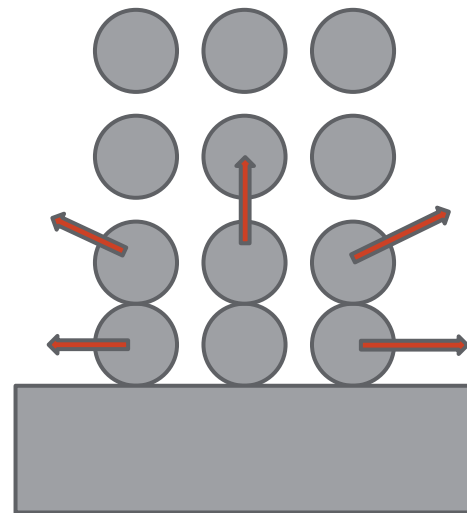




compression



pressure



force

Apply pressure

Apply additional forces(1)

If a particle is viscous, mix velocity with its contacting particles

$$v[i] \leftarrow v[i] + \Delta t * \mu * (v[j] - v[i])$$

μ : constant coefficient of viscosity

Apply additional forces(2)

If a particle is springy, apply force to restore distance between neighboring particles

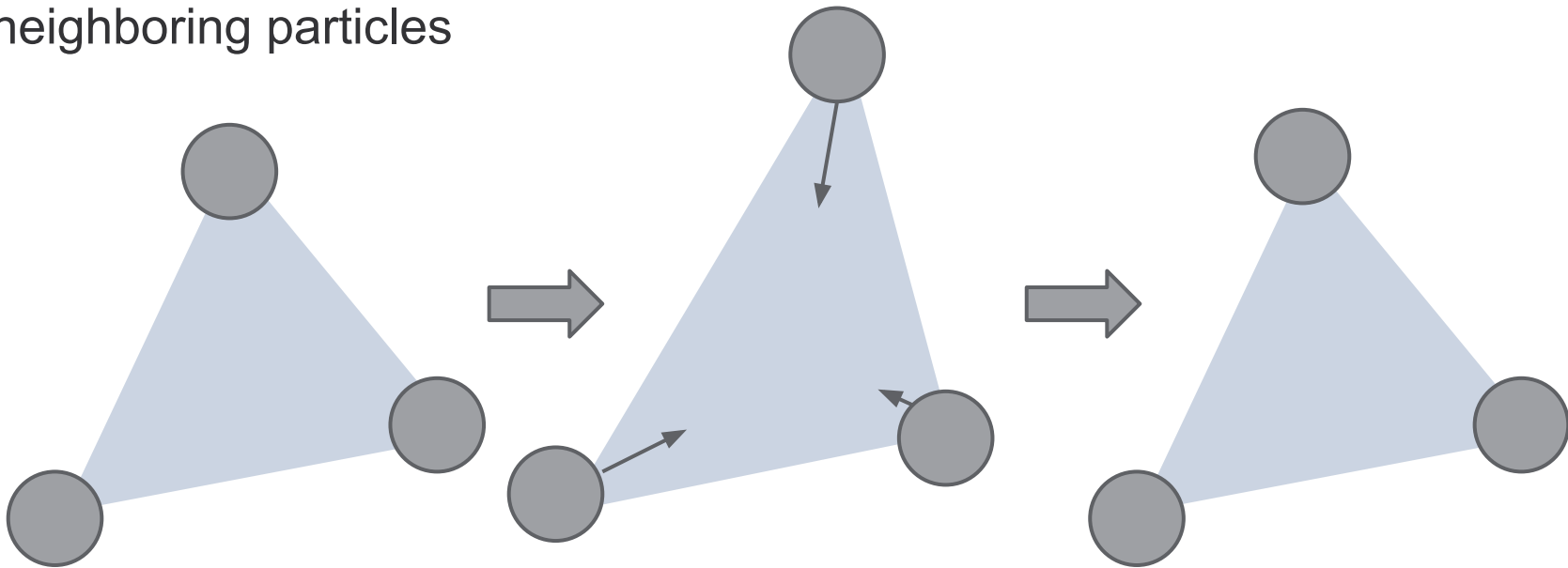
$$v[i] \leftarrow v[i] + \Delta t * k * (\text{distance}(p[j], p[i]) - L[i, j]) * n[i, j]$$

k: constant coefficient

L[i, j]: initial distance

Apply additional forces(3)

If a particle is elastic, apply force to restore relative positions among neighboring particles



Apply additional forces(4)

If a particle is powder, apply simple potential force instead of pressure

$$v[i] \leftarrow v[i] + \Delta t * k * \max(0, w[i,j] - w1) * n[i,j]$$

k: constant coefficient

w1: constant threshold weight

Apply additional forces(4)

If a particle is tensile, apply force below

$$w[i] = \text{sum}(w[i,j], j)$$

$$s[i] = \text{sum}((1-w[i,j]) * w[i,j] * n[i,j], j)$$

$$v[i] \leftarrow v[i] - \Delta t * (a * (w[i] + w[j] - 2 * w_0) + b * \text{dot}(s[j] - s[i], n[i,j])) * n[i,j]$$

a,b: constant coefficient

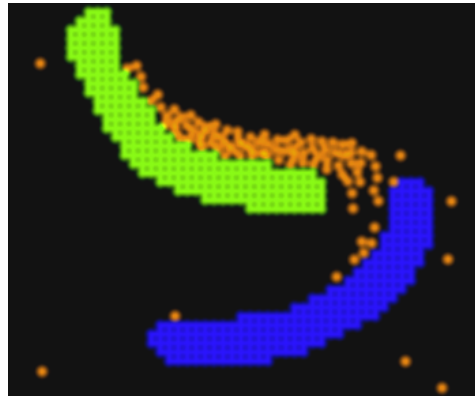
w₀: constant average weight



Restrict particle velocity(1)

For wall particles, set velocity to zero.

$$v[i] \leftarrow 0$$



Restrict particle velocity(2)

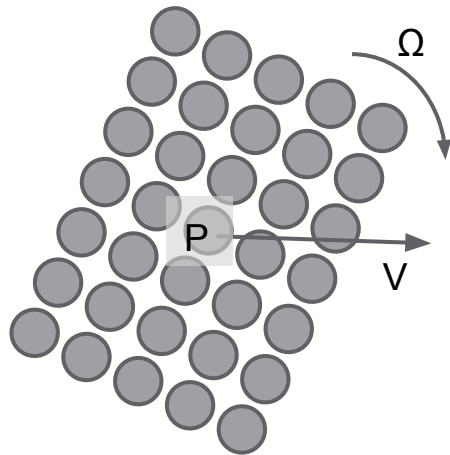
For particles in a rigid group, set velocities to maintain relative positions.

$$v[i] \leftarrow V + \text{cross}(\Omega, p[i] - P)$$

V : linear velocity of the group

Ω : angular velocity of the group

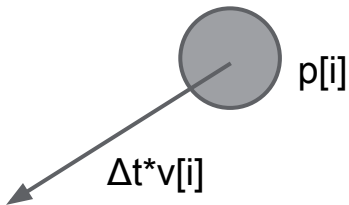
P : center of gravity of the group



Move particles

Add velocity multiplied by delta time to position for each particle

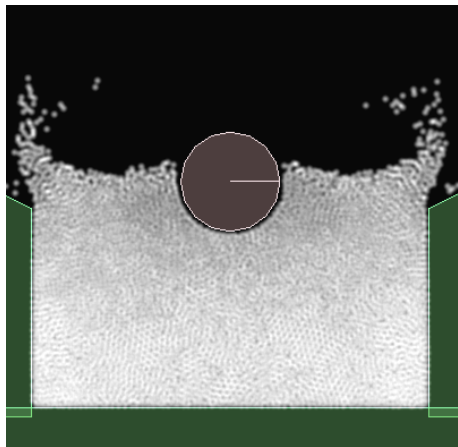
$$p[i] \leftarrow p[i] + \Delta t * v[i]$$

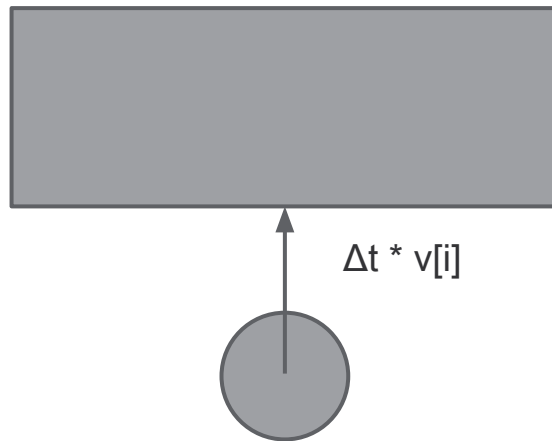
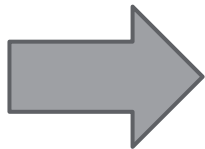
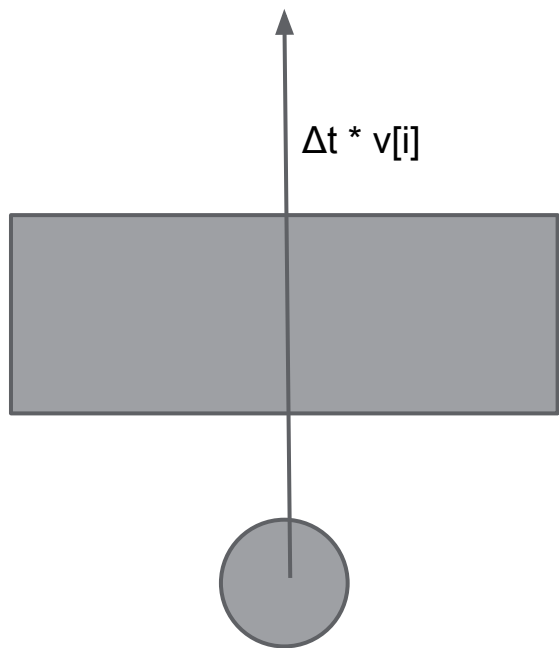


Interaction with Box2D (1)

Prevent penetration:

If a particle is about to penetrate a Box2D body during this step,
stop the particle at the surface



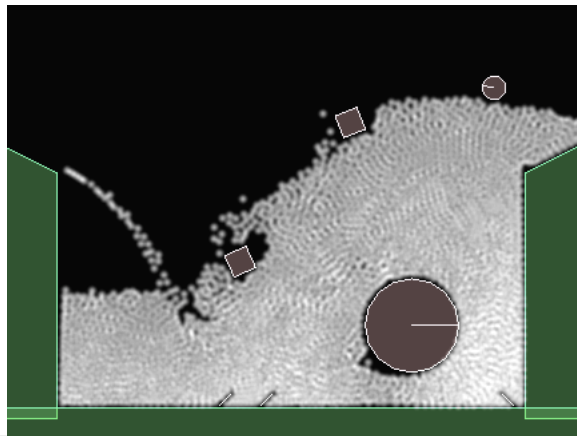


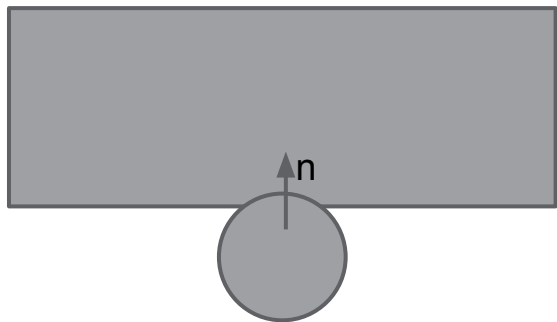
Prevent penetration

Interaction with Box2D (2)

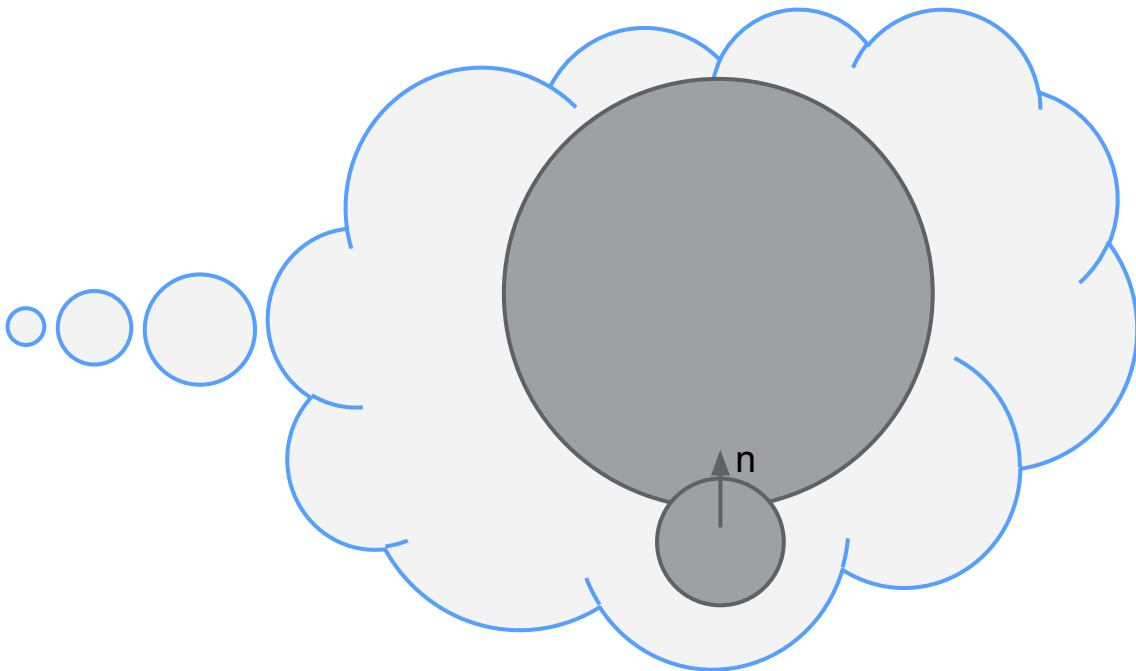
Continuous contact:

If a particle is in contact with a Box2D body, execute the subsequent particle simulation as if the particle is in contact with another particle





Continuous contact



contact with a pretend particle